COMBINATORIAL REALIZABILITY MODELS OF TYPE THEORY

PIETER HOFSTRA AND MICHAEL A. WARREN

ABSTRACT. We introduce a new model construction for Martin-Löf intensional type theory, which is sound and complete for the 1-truncated version of the theory. The model formally combines the syntactic model with a notion of realizability; it also encompasses the well-known Hofmann-Streicher groupoid semantics. As our main application, we use the model to analyse the syntactic groupoid associated to the type theory generated by a graph G, showing that it has the same homotopy type as the free groupoid generated by G.

CONTENTS

1. Introduction

This paper is a contribution to the study of the compelling connections between homotopy theory and Martin-Löf's intensional type theory (see [1] for a description of this programme). We present a new class of models for intensional type theory which allows us to gain insight into the homotopy-theoretic behaviour of the type theory in a way which is not possible with other models such as the syntactic model or the Hofmann-Streicher groupoid models.

We call these models *combinatorial realizability models*, because they associate to the syntactic data of the theory notions of *realizer*, and because realizers of terms of higher type are defined in terms of realizers of lower type, much in the same way as Kleene realizability defines realizers of universally quantified formulae in terms of functions sending realizers to realizers. (The extension of the notion of realizer to higher types is also closely resemblant of —and indeed inspired by— Tait's technique of logical predicates, see [7].) Unlike in classical realizability however, realizers have, a priori, nothing to do with computable functions; rather, realizers in our models are generally of a more combinatorial nature. Indeed, in our motivating example, realizers will be edges in a suitable graph; they can also be purely syntactical entities. In the limiting case where realizers are trivial (i.e. where every derivable term is trivially realized) the model reduces to the syntactic model, or rather a one-dimensional version thereof.

The type theories for which the model construction primarily is designed are dependent type theories having dependent sums and products with identity types which are 1-truncated (in the sense that higher identity proofs are forced to be definitional equalities, see below). The model also works for 0-truncated (i.e., extensional) type theories. The theories may further be assumed to have a natural number type with the usual recursion principle, and may further be extended by axioms postulating new basic types and terms. However, adding axioms involving non-basic types voids the warranty. We also have not investigated whether the model works in the presence of W-types and universes.

1.1. **Motivating problem: homotopy types of ML-complexes.** Let us describe in some detail the questions which prompted the investigations reported on here. Let G be a (directed, reflexive) graph and consider the theory $\mathbb{T}_1[G]$ obtained by augmenting ordinary intensional Martin-Löf type theory with the following data:

- A new basic type $\ulcorner G \urcorner$;
- A new basic term $\ulcorner a \urcorner : \ulcorner G \urcorner$ for each vertex a of G;
- A new basic term $\ulcorner f \urcorner : \underline{\ulcorner G \urcorner}(\ulcorner a \urcorner, \ulcorner b \urcorner)$ for each edge f with source a and target b;
- Axioms

$$\overline{\mathtt{r}(a) = \ulcorner 1_a \urcorner : \underline{\ulcorner G \urcorner}(\ulcorner a \urcorner, \ulcorner a \urcorner)}$$

  for each vertex a in G; and
- The *1-truncation rule* which states that iterated identity types are trivial:

2

$$\frac{p : \underline{A}(a,b)(f,g)}{f = g : \underline{A}(a,b)}$$

for any type A and terms $a, b : A$.

(We use the notation $\underline{A}(a,b)$ rather than $\mathtt{Id}_A(a,b)$ to denote the identity type at $a, b : A$ in order to stress the category-theoretic intuition and to make notation for iterated identity types more palatable.)

It follows from [6] that each closed type A in this theory has an associated groupoid $|A|$ with (definitional equality classes of) closed terms of type A as objects and where the hom-set $|A|(a,b)$ consists of closed terms of type $\underline{A}(a,b)$. The composition, identities and inverses of this groupoid are given by type-theoretic operations; by virtue of the 1-truncation rule the groupoid laws hold on the nose, and not just up to higher identity terms.

In particular, we may consider the syntactic groupoid $|\ulcorner G \urcorner|$ (which we simply write as $|G|$ from now on). Sending a graph G to the underlying graph of $|G|$ constitutes the object part of a monad T on the category **Graph** of (directed, reflexive) graphs; following [2], the algebras are called *1-truncated Martin-Löf complexes*. Understanding the behavior of these algebras is a first step in the project initiated in *ibid* of modeling homotopy types using the higher-categorical structures arising from type theory.

The first observation is that the theory $\mathbb{T}_1[G]$ admits an interpretation using the Hofmann-Streicher groupoid semantics: in order to specify such an interpretation, it suffices to interpret the basic data generating the theory. In principle, we can use any groupoid H and any graph morphism $G \to H$ to do this, but an obvious choice of H is of course the free groupoid $\mathcal{F}(G)$ on the graph G. For the interpretation of the basic terms of the theory we then may use the inclusion of generators $G \to \mathcal{F}(G)$. This completely determines the model.

By virtue of the interpretation of elimination terms for identity types, this yields in particular a functor

$$\Psi : |G| \to \mathcal{F}(G).$$

There is also a functor $\Phi : \mathcal{F}(G) \to |G|$ in the other direction, induced by the universal property of the free groupoid. It sends an object of $\mathcal{F}(G)$, i.e. a vertex a of G, to the basic term $\ulcorner a \urcorner : \ulcorner G \urcorner$, and a formal composite of edges of G to the type-theoretic composite. By the universal property of $\mathcal{F}(G)$, Φ is actually a section of Ψ.

We next note that the syntactic groupoid $|G|$ is, intuitively speaking, much larger than $\mathcal{F}(G)$. This is due to the fact that the type theory derives many more terms than just the generating basic terms coming from the graph G. For example, if f is a loop in the graph G on a vertex a and b is any vertex of G, the elimination rule for identity types gives

$$\frac{x, y : G, z : \underline{G}(x,y) \vdash G \qquad x : G \vdash b : G}{\mathtt{J}([x : G]b, a, a, f) : G}$$

where the first two hypotheses are simply obtained by weakening. One would like to know that such "duplicate" or "doppelgänger" terms do nothing homotopically harmful. For example, $T(G)$ should have the same connected components as G. Similarly, from a logical point of view one would like to know that there are no non-standard terms of natural number type in $\mathbb{T}_1[G]$ in the sense that we would like to prove that for each term t of natural number type there exists a numeral $\mathtt{n}$ and a term of type $\underline{N}(t, \mathtt{n})$. (Whether there exist non-standard terms of natural number type in the presence of Voevodsky's *univalence axiom* is a related question and we expect that the techniques developed in this paper can be modified to yield a proof of *Conjecture 1* from [8].)

The main application of our realizability model then, is to answer these and related questions. More concretely, one of the things we shall show is that the comparison functor $\Psi : |G| \to \mathcal{F}(G)$ is in fact an equivalence of groupoids. This is done by declaring a realizer of a closed term $t : G$ to be a morphism $t \to \bar{t}$ in $|G|$, where $\bar{t}$ is a basic term, i.e. a vertex of G. These realizers fit together

to form a natural transformation $1_{|G|} \Rightarrow \Phi\Psi$, exhibiting $\mathcal{F}(G)$ as a deformation retract of $|G|$. In particular, this proves that the two have the same homotopy type. It is shown in [2] that the techniques developed here can be used to prove that the category of 1-truncated ML-complexes is Quillen equivalent to the category of groupoids, and hence that 1-truncated ML-complexes model homotopy 1-types.

1.2. **Organization of the paper.**

Because the actual construction of the model and the proofs are somewhat technical and lengthy, we begin in Section 2 with an informal explanation of the construction. In particular, we explain what we mean by realizers, and how different suitable choices of realizers result in models which give us useful information about the syntax and the homotopy-theoretic behaviour of the type theory. We also explain in this section the general setup of the model using categories of families.

In Section 3 we develop various syntactical constructions which allow us to form a syntactic model of the type theory (but in a manner different from the ordinary term model). In particular, we show how to associate groupoids and pseudo-functors to types and contexts, and introduce some machinery for handling weakening and substitution on the level of syntactical groupoids. The main consequence of the technology introduced is now that the syntactic groupoids and pseudo-functors associated to the type theory give rise to a category with families.

Section 4 contains the main material. The realizability model itself is an augmentation of the syntactic model introduced in Section 3, and is obtained by gluing in a notion of realizability. We describe the general structure of this category with families, and then turn to the semantic type formers, dealing with dependent products, sums, identity types and natural numbers successively.

Details regarding some of the intuitively clear but technically involved type-theoretic constructions have been collected in Appendix B; finally, for ease of reference Appendix C summarizes the realizability clauses which can be extracted from the model construction in Section 4.

1.3. **Notational Conventions.**

The formulation of the rules of type theory we use are listed in complete detail in the Appendix. When it does not result in confusion, we suppress some typing information to reduce clutter. For example, instead of the cumbersome $\mathtt{J}_{[x,y:A,z:\underline{A}(x,y)]}([x : A]\phi, a, b, f)$, we simply write $\mathtt{J}(\phi, a, b, f)$ (or $\mathtt{J}([x : A]\phi, a, b, f)$ when we wish to make clear which variable is bound in this term).

Because we will often be dealing with lists of terms or variables it will be convenient to introduce a notation for such lists which will not result in excess clutter. In particular, we will denote such lists by using a bold face font. E.g., the list $x_1, \ldots, x_n$ of variables will be denoted by $\boldsymbol{x}$. Similar notation will be employed for lists of similar terms. For example, $\mathtt{r}(\boldsymbol{\alpha})$ denotes the list of terms $\mathtt{r}(\alpha_1), \ldots, \mathtt{r}(\alpha_n)$ where the list $\boldsymbol{\alpha}$ is understood. Also, $t[\boldsymbol{a}/\boldsymbol{x}]$ denotes the term $t[a_1/x_1, \ldots, a_n/x_n]$ and *not* the simultaneous substitution $t[a_1, \ldots, a_n/x_1, \ldots, x_n]$. We will often also avoid displaying variables when the context is clear.

2. Informal description of the model

This section explains the structure of the model without giving proofs. We first discuss the parameters in the model construction, namely the notion of realizer of terms of ground type. Various different applications of the model construction arise by choosing a suitable notion of realizer. We then proceed to explain the categorical form which the interpretation takes, and how terms of higher and dependent types are interpreted.

2.1. **Realizers and dense terms.**

We focus on a theory of the form $\mathbb{T}_1[G]$, where G is a graph. To specify an interpretation of such a theory, it will suffice to define what is meant by a realizer of a closed term of type G. Moreover, this assignment of realizers to closed terms has to be functorial,

in the sense that when $f : \underline{G}(a,b)$ is a closed term of identity type, we have an operation which sends realizers of a to realizers of b. Formally, this data amounts to a presheaf

$$\Vdash : |G| \to \mathbf{Set}$$

on the syntactic groupoid $|G|$. We suggestively write $\alpha \Vdash t : G$ instead of $\alpha \in \Vdash(t)$. Since the soundness theorem for the interpretation states that every term has a realizer, we must insist that basic terms have realizers. More precisely, for a notion of realizer $\Vdash$ to give rise to a sound interpretation, we must ask that there exists, for each vertex a of the graph G, a realizer $\alpha_a \Vdash a : G$, such that the functorial action of $\Vdash$ respects this, in the sense that for each edge $f : a \to b$ in G, $\alpha_a \cdot f = \alpha_b$ (where we write $\alpha \cdot f$ for the action of $\Vdash$ on arrows). Once this data has been specified, the model associates to each closed term $t : G$ a distinguished realizer.

In our motivating example, we are concerned with the functors $\Phi : \mathcal{F}(G) \to |G|$ and $\Psi : |G| \to \mathcal{F}(G)$ relating the syntactic groupoid $|G|$ on G to the free groupoid on G. The composite $\Phi\Psi : |G| \to |G|$ will be referred to as the *closure functor*, and will be denoted $t \mapsto \bar{t}$. We shall call a term t *dense* when its interpretation in the groupoid model based on the free groupoid $\mathcal{F}(G)$ is an identity. In particular, a closed term $f : \underline{G}(a,b)$, regarded as a morphism in the syntactic groupoid $|G|$, is dense whenever its closure is an identity $\bar{f} = 1_{\bar{a}}$.

With this notation and terminology, we now define a realizer of a closed term $a : G$ to be a dense map $\alpha : a \to \bar{a}$. We must also specify how this assignment is functorial in a: given a morphism $f : \underline{G}(a,b)$ and a realizer $\alpha : a \to \bar{a}$, consider the composite

$$b \xrightarrow{f^{-1}} a \xrightarrow{\alpha} \bar{a} \xrightarrow{\bar{f}} \bar{b}$$

Since the closure of α is the identity, it is easily seen that this morphism is dense, i.e. that it is a realizer of b. It is also clear that this is functorial.

What is not a priori obvious however is that all closed terms have realizers. Indeed, this is the content of the soundness lemma. However, basic terms are trivially realized by the identity.

It is not necessary, for the specification of a realizability model, to state what realizers of terms of identity types $\underline{G}(a,b)$ are. Indeed, assuming we have realizers $\alpha : a \to \bar{a}$ and $\beta : b \to \bar{b}$, we shall simply declare a closed term $f : \underline{G}(a,b)$ to be realized when the square

$$
\begin{array}{ccc}
a & \xrightarrow{f} & b \\
\alpha \downarrow & & \downarrow \beta \\
\bar{a} & \xrightarrow{\bar{f}} & \bar{b}
\end{array}
$$

commutes, i.e. when β is the result of reindexing the realizer α along f.

Remark. An obvious generalization suggests itself: instead of taking realizers for identity proofs between terms to be determined by those of the terms themselves (and the functorial action on realizers) we could take a realizer to be a higher identity proof witnessing the fact that the above square commutes up to equivalence. We conjecture that this is the appropriate generalization of the model to theories without the truncation rule.

There are other notions of realizer which may be used, and in order to avoid the impression that the above example is the only one, we briefly point out a couple of other possibilities (several other examples can be found in [2]). First, one could consider the terminal presheaf $|G| \to \mathbf{Set}$ which sends each object to a fixed singleton set. In this case, realizers contain no information, and the resulting model will simply be a 1-truncated syntactic model (i.e. a model of the theory purely based on syntactic groupoids). Second, more general closure functors can be used. Any

endofunctor $F : |G| \to |G|$ gives rise to a model in which a realizer for a term a is a dense map $a \to Fa$, as long as basic terms have realizers.

A further generalization is possible:

Theorem 2.1. *Let G be a graph, H a groupoid and let $P, Q : |G| \to H$ be two functors. Suppose furthermore that we are given a morphism $\alpha_a : P(a) \to Q(a)$ for each basic term a such that, for each basic term $f : a \to b$, the following diagram commutes:*

$$
\begin{array}{ccc}
P(a) & \xrightarrow{\ \alpha_a\ } & Q(a) \\
{\scriptstyle P(f)}\big\downarrow & & \big\downarrow{\scriptstyle Q(f)} \\
P(b) & \xrightarrow[\ \alpha_b\ ]{} & Q(b).
\end{array}
$$

Then there exists a natural transformation $\alpha : P \Rightarrow Q$ whose component at a basic term a is α_a.

Proof. We define a notion of realizability for G: let $\Vdash : |G| \to \mathbf{Set}$ be the presheaf which sends an object $t : G$ to the set $H(P(t), Q(t))$. The functorial action is by conjugation: given $f : t \to s$ and $\tau \Vdash t : G$, we set $\tau \cdot f =_{\mathrm{def}} Q(f)\tau P(f)^{-1}$. A basic term $a : A$ is then realized by the given α_a. This gives a realizability model, the soundness of which gives in particular that each $t : G$ has a realizer, and that these form a natural transformation as desired. $\qquad\square$

An application of this more general result appears in [2], where it is used to show that the free 1-truncated ML-complex on a groupoid G (i.e. an object in the image of the left adjoint to the forgetful functor from ML-complexes to groupoids) is equivalent to G.

2.2. **The general setup of the model.**
We now turn to the general organization of the model, which augments the syntactic model with the concept of realizer. We begin by sketching the syntactic structure, and then explain how the realizers are added into this model.

As above, we shall denote the syntactic groupoid associated to a closed type A by $|A|$. This can be extended to contexts and type judgements: to each well-formed context Γ we associate a groupoid $|\Gamma|$, and to each type judgement $\Gamma \vdash T(\boldsymbol{x})$ we associate a pseudo-functor

$$|\Gamma \vdash T(\boldsymbol{x})| : |\Gamma| \to \mathbf{Gpd}$$

which sends an object $\boldsymbol{c}$ of $|\Gamma|$ to the syntactic groupoid $|T(\boldsymbol{c})|$. (The action on morphisms is given by a type-theoretic version of change-of-base; details will be provided in section 3.) Then we use the Grothendieck construction to define the syntactic groupoid of the extended context to be

$$|\Gamma, y : T(\boldsymbol{x})| = \int |\Gamma \vdash T(\boldsymbol{x})|.$$

Thus we obtain a fibration of groupoids

$$|\Gamma, y : T| \to |\Gamma|.$$

Suppose now that we have a term judgement $\Gamma \vdash t : T$. Then the interpretation $|\Gamma \vdash t : T|$ of t will be a section of the fibration $|\Gamma, y : T| \to |\Gamma|$, namely the one which sends an object $\boldsymbol{c}$ of Γ to the object $t(\boldsymbol{c})$.

The realizability model will interpret the syntax in a similar way. We will denote the interpretation in the realizability model by $[\![-]\!]$ (where it is assumed we have fixed a notion of realizer).

First, consider the basic type $\ulcorner G \urcorner$. By definition, a notion of realizer is a functor $\mathcal{R} : |G| \to \mathbf{Set}$. Applying the Grothendieck construction to this presheaf, we obtain a groupoid

$$[\![G]\!] = \int \mathcal{R}$$

which we take to be the interpretation of the type G. It naturally comes with a fibration

$$\llbracket G \rrbracket \xrightarrow{\pi_G} |G|.$$

This will be a general pattern: the realizability interpretation of any closed type (context) Γ will be a groupoid $\llbracket \Gamma \rrbracket$ fibred over $|\Gamma|$:

$$\llbracket \Gamma \rrbracket \xrightarrow{\pi_\Gamma} |\Gamma|.$$

It is technically convenient to introduce an intermediate construction: to each context Γ we will also associate a groupoid $\|\Gamma\|$. Thus we will have three groupoids $|\Gamma|, \|\Gamma\|$ and $\llbracket \Gamma \rrbracket$ which will fit together in a commutative diagram

$$\begin{array}{ccc} \llbracket \Gamma \rrbracket & \longrightarrow & \|\Gamma\| \\ & \llap{\pi_\Gamma} \searrow & \downarrow \\ & |\Gamma| & \end{array} \tag{1}$$

such that each of the components is a fibration. With this picture in mind, the interpretation of a judgement $\Gamma \vdash T$ is a functor

$$\llbracket \Gamma \vdash T \rrbracket : \|\Gamma, x : T\| \to \mathbf{Set}$$

which is to be thought of as sending an object to the set of realizers of that object; then the extended context is interpreted by Grothendieck construction as

$$\llbracket \Gamma, x : T \rrbracket =_{\mathrm{def}} \int \llbracket \Gamma \vdash T \rrbracket.$$

We will also have at each stage of the construction that

$$\|\Gamma, x : T\| =_{\mathrm{def}} \int |\Gamma \vdash T| \circ \pi_\Gamma$$

so that there is always a pullback diagram

$$\begin{array}{ccc} \|\Gamma, x : T\| & \longrightarrow & |\Gamma, x : T| \\ \downarrow & & \downarrow \\ \llbracket \Gamma \rrbracket & \xrightarrow{\pi_\Gamma} & |\Gamma| \end{array}$$

in the category of groupoids.

Putting this together with what was described in (1) above we have

$$\begin{array}{ccc} \llbracket \Gamma, x : T \rrbracket & \xrightarrow{\pi_{(\Gamma, x:T)}} & \\ \downarrow & & \\ \|\Gamma, x : T\| & \longrightarrow & |\Gamma, x : T| \\ \downarrow & & \downarrow \\ \llbracket \Gamma \rrbracket & \xrightarrow{\pi_\Gamma} & |\Gamma| \end{array}$$

The intuition to keep in mind here is that the groupoid $\|\Gamma, x : T\|$ has objects $(\boldsymbol{c}, \boldsymbol{\gamma}, t)$, where $\boldsymbol{c}$ is an object of the syntactic groupoid Γ, where the $\boldsymbol{\gamma}$ are realizers for $\boldsymbol{c}$, and where t is an object of $|\Gamma, x : T|$ over $\boldsymbol{c}$. Thus in this intermediate groupoid there is no information about realizers for t included, and this is precisely the difference between $\|\Gamma, x : T\|$ and the full interpretation $\llbracket \Gamma, x : T \rrbracket$, where an object is of the form $((\boldsymbol{c}, \boldsymbol{\gamma}), t, \tau)$, with τ a realizer for t.

Indeed, in order to make this intuition a bit more apparent we will often write

$$\tau \Vdash_{\boldsymbol{c},\boldsymbol{\gamma}} t : T(\boldsymbol{c})$$

to indicate that τ is an object of the (discrete) groupoid $[\![\Gamma \vdash T]\!]_{(\boldsymbol{c},\boldsymbol{\gamma},t)}$. When no confusion will result we often drop $\boldsymbol{c}$ from the subscript and simply write $\tau \Vdash_{\boldsymbol{\gamma}} t : T(\boldsymbol{c})$. Functoriality of $[\![\Gamma \vdash T]\!]$ implies that if τ is as above and $(\boldsymbol{h}, m) : (\boldsymbol{c}, \boldsymbol{\gamma}, t) \to (\boldsymbol{c}', \boldsymbol{\gamma}', t')$ is a morphism in $\|\Gamma, x : T\|$, then we have

$$\tau \cdot (\boldsymbol{h}, m) \Vdash_{\boldsymbol{c}',\boldsymbol{\gamma}'} t' : T(\boldsymbol{c}').$$

Thus, in order to give the interpretation of a type judgement $\Gamma \vdash T$ it suffices to provide the following data:

- We must say what is a realizer

$$\tau \Vdash_{\boldsymbol{\gamma}} t : T(\boldsymbol{c})$$

for $(\boldsymbol{c}, \boldsymbol{\gamma})$ an object of $[\![\Gamma]\!]$ and $t : T(\boldsymbol{c})$ a term; and
- We must give a reindexing action which sends a realizer $\tau \Vdash_{\boldsymbol{\gamma}} t : T(\boldsymbol{c})$ to

$$\tau \cdot (\boldsymbol{h}, m) \Vdash_{\boldsymbol{\gamma}'} t' : T(\boldsymbol{c}')$$

for any arrow $(\boldsymbol{h}, m) : (\boldsymbol{c}, \boldsymbol{\gamma}, t) \to (\boldsymbol{c}', \boldsymbol{\gamma}', t')$ in $\|\Gamma, x : T\|$, and we must verify functoriality of reindexing.

2.3. **Interpretation of terms.** A term $\Gamma \vdash t : T$ will be interpreted as a section

satisfisfying the condition that the following diagram

commutes. This means that to give the interpretation of a term $\Gamma \vdash t : T$ it suffices to give, for each object $(\boldsymbol{c}, \boldsymbol{\gamma})$ of $[\![\Gamma]\!]$, a realizer[1]

$$t[\boldsymbol{\gamma}] \Vdash_{\boldsymbol{\gamma}} t(\boldsymbol{c}) : T(\boldsymbol{c}),$$

and to prove that

$$t[\boldsymbol{\gamma}] \cdot \big(\boldsymbol{h}, t|\boldsymbol{h}\big) = t[\boldsymbol{\gamma}'] \tag{2}$$

for each arrow $\boldsymbol{h} : (\boldsymbol{c}, \boldsymbol{\gamma}) \to (\boldsymbol{c}', \boldsymbol{\gamma}')$ in $[\![\Gamma]\!]$. Here, $t|\boldsymbol{h} : t(\boldsymbol{c}) \to t(\boldsymbol{c}')$ denotes the (second component of) the action of t on the morphism $\boldsymbol{h}$, see section 3 for details.

---

[1]I.e., $t[\boldsymbol{\gamma}]$ is the realizer part of $[\![\Gamma \vdash t : T]\!](\boldsymbol{c}, \boldsymbol{\gamma})$: $[\![\Gamma \vdash t : T]\!](\boldsymbol{c}, \boldsymbol{\gamma}) = (\boldsymbol{c}, \boldsymbol{\gamma}, t(\boldsymbol{c}), t[\boldsymbol{\gamma}])$.

2.4. **Categories with Families.** In order to ensure that the interpretations of contexts, types and terms is independent of derivations, we organize our semantics as a *Category with Families* (see [3, 5] for precise definitions, examples, and connections to other notions of semantics for dependent type theories). Such a model consists of a category of semantic contexts (which will serve as interpretations of contexts) and semantic context morphisms (which will serve as interpretations of context morphisms). In addition, to each semantic context C there is associated a set of semantic types $Ty(C)$ in a contravariant manner; and to each semantic type $T \in Ty(C)$ there is associated a set of semantic terms $Tm(T)$, again with a contravariant action by context morphisms. Finally, one requires that this structure admits *comprehension*, in the usual fibrational sense.

A category with families (CwF from now on) interprets the basic calculus of type dependency; definitionally equal entities in the type theory are interpreted as set-theoretically equal entities in the model. We shall also need to interpret dependent products, dependent sums, natural numbers and intensional identity types. A CwF is said to support dependent products (dependent sums, natural numbers, identity types) if it admits the categorical counterparts of the type and term constructors for dependent product types (resp. dependent sums, natural numbers, identity types).

We will actually construct our model in two stages. First, we show that there is a CwF of syntactic groupoids: this we obtain by taking the syntactic model (whose underlying category is the category of contexts) and then transfer this model along the functor which takes a context Γ and returns its syntactic groupoid $|\Gamma|$.

Then we form the gluing of this functor and consider the full subcategory on the cloven fibrations. An object will have the form $\llbracket \Gamma \rrbracket \to |\Gamma|$, and this is precisely how the realizability semantics interprets a context Γ. Types over Γ are then defined to be extended contexts $\Gamma, x : A$, together with a specification of realizers $\|\Gamma, x : A\| \to \mathbf{Set}$.

3. The syntactic structure

We now describe in detail the syntactic part of the model. First, we introduce sequential J-terms, which are highly useful in describing the type-theoretic constructions corresponding to the iterated Grothendieck constructions appearing in the interpretation of contexts and types. We then explain the syntactic interpretation of types and terms and establish soundness.

3.1. **Sequential J-terms.** To begin, recall the type-theoretic analogue of (covariantly) reindexing a term $t : B(a)$ along an identity proof $f : \underline{A}(a, b)$ (where $x : A \vdash B(x)$): we define the term $t * f : B(b)$ as

$$
\frac{
  \begin{array}{c}
    x, y : A, z : \underline{A}(x, y) \vdash B(y)^{B(x)} \\
    x : A \vdash \lambda v.v : B(x)^{B(x)} \\
    \vdash f : \underline{A}(a, b) \\
    \hline
    \vdash \mathtt{J}(\lambda v.v, a, b, f) : B(b)^{B(a)} \qquad \vdash t : B(a)
  \end{array}
}{
  \vdash \mathtt{app}(\mathtt{J}(\lambda v.v, a, b, f), t) : B(b)
}
$$

Note that in the groupoid interpretation the judgement $x : A \vdash B(x)$ takes the form of a functor from the groupoid interpreting the context $x : A$ to the category of groupoids. A term $f : \underline{A}(a, b)$ is interpreted as an arrow $f : a \to b$ in the groupoid interpreting $x : A$, and the term $t * f$ is then interpreted as $t \cdot f$, where we have omitted semantic brackets in order to simplify notation and where $- \cdot f$ denotes the action of the functor interpreting $x : A \vdash B(x)$ on f.

Of course, this reindexing action does not just work for closed terms; the above derivation also works in an ambient context; in particular, both t and a, b and f can simply be variables.

Given a context

$$
\Gamma = \big(x_1 : A_1, \ldots, x_n : A_n \big)
$$

there is a new context $\tilde{\Gamma}$ which consists of variable declarations that we now describe. Thinking of the term forming operation $- * -$ described above as base change the context $\tilde{\Gamma}$ should be thought of encoding the type theoretic data corresponding to arrows in the Grothendieck construction of the associated functor. The first part of $\tilde{\Gamma}$ has variable declarations

$$\big(x_1, y_1 : A_1, x_2 : A_2(x_1), y_2 : A_2(y_1), \ldots, x_n : A_n(\boldsymbol{x}), y_n : A_n(\boldsymbol{y})\big).$$

It then consists of a variable declaration $z_1 : \underline{A_1}(x_1, y_1)$. Given such a variable declaration we have $x_2 * z_1 : A_2(y_1)$. As such, we may form the next variable declaration $z_2 : \underline{A_2(y_1)}(x_2 * z_1, y_2)$. Similarly, given z_2 we may form $x_3 * z_1 : A_3(y_1, x_2 * z_1)$ and then $(x_3 * z_1) * z_2 : \underline{A_3(y_1, y_2)}$. Accordingly, the next variable declaration that we add is of the form

$$z_3 : \underline{A_3(y_1, y_2)}\big((x_3 * z_1) * z_2, y_3\big).$$

So far, the variables z_1, z_2 and z_3 may be thought of as representing an arrow from x_3 to y_3 in the groupoid associated to the context $x_1 : A_1, x_2 : A_2(x_1), x_3 : A_3(x_1, x_2)$. In a diagram:

where z_3 is an arrow in the fibre over y_2 and z_2 an arrow in the fibre over y_1.

In general, for each $1 < m \le n$,

$$z_m : \underline{A_m(y_1, \ldots, y_{m-1})}\big((\cdots(x_m * z_1) * \cdots) * z_{m-1}, y_m\big)$$

is an arrow in the fibre over y_{m-1}. Then, $\tilde{\Gamma}$ is the collection of all of the variable declarations

$$x_1, \ldots, x_n, y_1, \ldots, y_n, z_1, \ldots, z_n$$

of the types described above. Note that we have

$$\tilde{\Gamma}[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}] = \Gamma.$$

We observe that we have the following derived rule

$$\frac{\tilde{\Gamma} \vdash T \qquad \Gamma \vdash \varphi : T[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}]}{\tilde{\Gamma} \vdash \mathsf{J}^{\sigma}_{[\tilde{\Gamma} \vdash T]}([\boldsymbol{x}]\varphi, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) : T}$$

Here, as in the usual formulation of the elimination rule, the variables enclosed in square brackets $[\boldsymbol{x}]$ indicate that these variables are bound in the term φ. When no confusion will result these and other bits of bookkeeping (such as the subscript $\tilde{\Gamma} \vdash T$) will be omitted.

Just as an ordinary J-term can be regarded as an expansion of a term along an arrow, a sequential J-term is to be thought of as an expansion of a term along an arrow in an iterated Grothendieck construction.

These terms satisfy the following conversion rule

$$\frac{\tilde{\Gamma} \vdash T \qquad \Gamma \vdash \varphi : T[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}]}{\Gamma \vdash \mathsf{J}^{\sigma}(\varphi, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}] = \varphi}$$

10

The explicit definition of these terms and the corresponding parameterized versions can be found in Section B.2. We point out that this construction is not novel: it appeared in [4].

3.2. Groupoids and pseudo-functors associated to types.

We now proceed to associate to each context Γ a groupoid $|\Gamma|$, and to each type judgement $\Gamma \vdash T$ a pseudo-functor

$$|\Gamma \vdash T| : |\Gamma| \to \mathbf{Gpd}.$$

This will be done by induction on the length of the context Γ.

When $\Gamma = ()$ is the empty context, we set $|()| = 1$, the terminal groupoid. For a closed type judgement $\vdash A$ (i.e. when A is a closed type) we set $| \vdash A| : 1 \to \mathbf{Gpd}$ to be the pseudo-functor corresponding to the groupoid $|A|$, the syntactic groupoid of A (which, as explained earlier, has definitional equality classes of closed terms $a : A$ as objects and terms $f : \underline{A}(a, b)$ as its morphisms). The context $(x : A)$ is then interpreted as the Grothendieck construction of this pseudo-functor, i.e. we have $|x : A| =_{\text{def}} |A|$.

Next, suppose that we have already defined the groupoid $|\Gamma|$ associated to a valid context Γ, and that we are given a type judgement $\Gamma \vdash T$. We wish to define the pseudo-functor $|\Gamma \vdash T| : |\Gamma| \to \mathbf{Gpd}$. On an object $\boldsymbol{c}$ of Γ, we define

$$|\Gamma \vdash T|_{\boldsymbol{c}} =_{\text{def}} |T(\boldsymbol{c})|.$$

That is, the object $\boldsymbol{c}$ is sent to the syntactic groupoid of the closed type $T(\boldsymbol{c})$. To an arrow $\boldsymbol{h} : \boldsymbol{c} \to \boldsymbol{d}$ in $|\Gamma|$ we associate the functor $|T(\boldsymbol{c})| \to |T(\boldsymbol{d})|$ given by

$$t : T(\boldsymbol{c}) \longmapsto \left(\cdots (t * h_1) \cdots * h_n \right) : T(\boldsymbol{d})$$

Note that this definition makes sense because, by hypothesis,

$$h_1 : \underline{C_1}(c_1, d_1)$$
$$h_2 : \underline{C_2(d_1)}(c_2 * h_1, d_2)$$
$$h_3 : \underline{C_3(d_1, d_2)}\big((c_3 * h_1) * h_2, d_3\big)$$
$$\vdots$$
$$h_n : \underline{C_n(d_1, \ldots, d_{n-1})}\big((c_n * h_1) \cdots * h_{n-1}, d_n\big).$$

We will usually denote the action of this functor by $- \cdot \boldsymbol{h}$. Given $\boldsymbol{h} : \boldsymbol{c} \to \boldsymbol{d}$ and $\boldsymbol{k} : \boldsymbol{d} \to \boldsymbol{e}$ we must construct the coherence natural isomorphism $\gamma(\boldsymbol{h}, \boldsymbol{k})$ indicated in the following diagram:

$$
\begin{array}{ccc}
|T(\boldsymbol{c})| & \xrightarrow{\ -\cdot(\boldsymbol{k} \circ \boldsymbol{h})\ } & |T(\boldsymbol{e})| \\
& \Downarrow & \\
{\scriptstyle -\cdot \boldsymbol{h}} \searrow & & \nearrow {\scriptstyle -\cdot \boldsymbol{k}} \\
& |T(\boldsymbol{d})| &
\end{array}
$$

For an object t of $|T(\boldsymbol{c})|$ the natural isomorphism $\gamma(\boldsymbol{h}, \boldsymbol{k})$ has component

$$\gamma(\boldsymbol{h}, \boldsymbol{k})_t : t \cdot (\boldsymbol{k} \circ \boldsymbol{h}) \to (t \cdot \boldsymbol{h}) \cdot \boldsymbol{k}$$

given by the (parameterized) sequential J-term

$$\mathsf{J}^\sigma\big(\mathtt{r}(u \cdot \boldsymbol{w}), \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{h}, \boldsymbol{e}, \boldsymbol{k}, t\big) : \underline{T(\boldsymbol{e})}\big(t \cdot (\boldsymbol{k} \circ \boldsymbol{h}), (t \cdot \boldsymbol{h}) \cdot \boldsymbol{k}\big)$$

Because $- \cdot 1_{\boldsymbol{c}}$ is not just isomorphic but equal to the identity, we do not need to specify a unit coherence isomorphism. Finally, the coherence laws follow from the 1-truncation axiom.

As a consequence of the above, we have the following explicit description of composition in the groupoid $|\Gamma, x : T|$:

$$(\boldsymbol{f}', h') \circ (\boldsymbol{f}, h) = \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ (h \cdot \boldsymbol{f}') \circ \gamma(\boldsymbol{f}, \boldsymbol{f}')\big).$$

11

We will often denote the second component of this map by $h' \circ_{\boldsymbol{f}, \boldsymbol{f}'} h$.

3.3. Interpretation of terms.

Now that we have an interpretation of types and contexts, we turn to the interpretation of terms. For an open term $\Gamma \vdash t : T$, the functor $|t| : |\Gamma| \to |\Gamma, y : T|$ sends an object $\boldsymbol{c}$ of $|\Gamma|$ to the object $(\boldsymbol{c}, t(\boldsymbol{c}))$ of $|\Gamma, y : T|$. An arrow $\boldsymbol{h} : \boldsymbol{c} \to \boldsymbol{d}$ of $|\Gamma|$ is sent by $|t|$ to the arrow $(\boldsymbol{h}, t|\boldsymbol{h}) : (\boldsymbol{c}, t(\boldsymbol{c})) \to (\boldsymbol{d}, t(\boldsymbol{d}))$ where $t|\boldsymbol{h}$ is the term

$$\frac{\tilde{\Gamma} \vdash \underline{T(\boldsymbol{y})}(t(\boldsymbol{x}) \cdot \boldsymbol{z}, t(\boldsymbol{y}))}{\dfrac{\Gamma \vdash \mathtt{r}(t(\boldsymbol{x})) : \underline{T(\boldsymbol{x})}(t(\boldsymbol{x}), t(\boldsymbol{x}))}{\mathtt{J}^{\sigma}(\mathtt{r}(t(\boldsymbol{x})), \boldsymbol{c}, \boldsymbol{d}, \boldsymbol{h}) : \underline{T(\boldsymbol{d})}(t(\boldsymbol{c}) \cdot \boldsymbol{h}, t(\boldsymbol{d}))}}$$

For a closed term $t : T$ we have the obvious global section $|t| : 1 \to |x : T|$.

We need to show that $|t|$ is actually a functor. Clearly, when $\boldsymbol{h}$ is the identity then by the conversion rule for sequential J-terms $t|\boldsymbol{h} = \mathtt{r}(t(\boldsymbol{c}))$. Given composable maps $\boldsymbol{c} \xrightarrow{\ \boldsymbol{h}\ } \boldsymbol{d} \xrightarrow{\ \boldsymbol{k}\ } \boldsymbol{e}$, we need to verify that the following square commutes:

$$
\begin{array}{ccc}
t(\boldsymbol{c}) \cdot \boldsymbol{h} \cdot \boldsymbol{k} & \xrightarrow{\ (t|\boldsymbol{k}) \cdot \boldsymbol{h}\ } & t(\boldsymbol{d}) \cdot \boldsymbol{k} \\
{\scriptstyle \gamma} \downarrow & & \downarrow {\scriptstyle t|\boldsymbol{k}} \\
t(\boldsymbol{c}) \cdot (\boldsymbol{k}\boldsymbol{h}) & \xrightarrow[\ t|\boldsymbol{h}\boldsymbol{k}\ ]{} & t(\boldsymbol{e}).
\end{array}
$$

However, there is a propositional equality between the two composites (take $\boldsymbol{k}$ to be the identity so that we get a definitional equality $t|\boldsymbol{h} = \mathtt{r}(t(\boldsymbol{d})) \circ (t|\boldsymbol{h})$, and then form the appropriate sequential J-term). Hence by 1-truncation the two are definitionally equal.

3.4. Weakening.

Consider a weakening inference

$$\frac{\Gamma \vdash A \qquad \Delta \vdash}{\Gamma, \Delta \vdash A}$$

Note that there exists a functor

$$\tau_{\Gamma; \Delta; A} : |\Gamma, \Delta, x : A| \to |\Gamma, x : A| \tag{3}$$

given as follows:

On Objects: For $(\boldsymbol{c}, \boldsymbol{d}, a)$ an object of $|\Gamma, \Delta, x : A|$ we set

$$\tau_{\Gamma; \Delta; A}(\boldsymbol{c}, \boldsymbol{d}, a) =_{\text{def}} (\boldsymbol{c}, a).$$

On Arrows: For an arrow $(\boldsymbol{f}, \boldsymbol{g}, h) : (\boldsymbol{c}, \boldsymbol{d}, a) \to (\boldsymbol{c}', \boldsymbol{d}', a')$ we first observe that there exists an arrow

$$a\dagger(\boldsymbol{f}, \boldsymbol{g}) : a \cdot \boldsymbol{f} = |\Gamma \vdash A|_{\boldsymbol{f}}(a) \to a \cdot (\boldsymbol{f}, \boldsymbol{g}) = |\Gamma, \Delta \vdash A|_{(\boldsymbol{f}, \boldsymbol{g})}(a)$$

given by the term

$$\frac{\tilde{\Delta} \vdash \underline{A(\boldsymbol{c}')}\big(a \cdot \boldsymbol{f}, a \cdot (\boldsymbol{f}, \boldsymbol{z})\big)}{\dfrac{\Delta \vdash \mathtt{r}(a \cdot \boldsymbol{f}) : \underline{A(\boldsymbol{c}')}(a \cdot \boldsymbol{f}, a \cdot \boldsymbol{f})}{\mathtt{J}^{\sigma}\big(\mathtt{r}(a \cdot \boldsymbol{f}), \boldsymbol{d}, \boldsymbol{d}', \boldsymbol{g}\big) : \underline{A(\boldsymbol{c}')}\big(a \cdot \boldsymbol{f}, a \cdot (\boldsymbol{f}, \boldsymbol{g})\big).}}$$

As such, we define $\tau_{\Gamma; \Delta; A}(\boldsymbol{f}, \boldsymbol{g}, h) = \big(\boldsymbol{f}, h \circ (a\dagger(\boldsymbol{f}, \boldsymbol{g}))\big)$ where $h \circ a\dagger(\boldsymbol{f}, \boldsymbol{g})$ is the composite indicated in the following diagram:

$$a \cdot \boldsymbol{f} \xrightarrow{\ a\dagger(\boldsymbol{f}, \boldsymbol{g})\ } a \cdot (\boldsymbol{f}, \boldsymbol{g}) \xrightarrow{\ h\ } a'$$

Clearly the term $a \dagger (\boldsymbol{f}, \boldsymbol{g})$ is definitionally equal to the reflexivity on a when $\boldsymbol{f}$ and $\boldsymbol{g}$ are reflexivities; if h is also a reflexivity then this implies that $\tau_{\Gamma;\Delta;A}(\boldsymbol{f}, \boldsymbol{g}, h)$ is one as well.

To see that composition is preserved, consider morphisms $(\boldsymbol{f}, \boldsymbol{g}, h) : (\boldsymbol{c}, \boldsymbol{d}, a) \to (\boldsymbol{c}', \boldsymbol{d}', a')$ and $(\boldsymbol{f}', \boldsymbol{g}', h') : (\boldsymbol{c}', \boldsymbol{d}', a') \to (\boldsymbol{c}'', \boldsymbol{d}'', a'')$. First, we have

$$\big(\boldsymbol{f}', \boldsymbol{g}', h'\big)\big(\boldsymbol{f}, \boldsymbol{g}, h\big) = \big(\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g}), h' \circ_{(\boldsymbol{f}, \boldsymbol{g}),(\boldsymbol{f}', \boldsymbol{g}')} h\big).$$

Next, observe the coherence isomorphism

$$\gamma((\boldsymbol{f}, \boldsymbol{g}), (\boldsymbol{f}', \boldsymbol{g}')) : a \cdot (\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g} \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g})) \to a \cdot (\boldsymbol{f}, \boldsymbol{g}) \cdot (\boldsymbol{f}', \boldsymbol{g}')$$

makes the following diagram commute

$$\begin{array}{ccccc}
a \cdot (\boldsymbol{f}'\boldsymbol{f}) & \xrightarrow{\gamma(\boldsymbol{f}, \boldsymbol{f}')} & a \cdot \boldsymbol{f} \cdot \boldsymbol{f}' & \xrightarrow{(a \cdot \boldsymbol{f})\dagger(\boldsymbol{f}', \boldsymbol{g}')} & a \cdot \boldsymbol{f} \cdot (\boldsymbol{f}', \boldsymbol{g}') \\
{\scriptstyle a\dagger(\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g})} \downarrow & & & & \downarrow {\scriptstyle (a\dagger(\boldsymbol{f}, \boldsymbol{g}))\cdot(\boldsymbol{f}', \boldsymbol{g}')} \\
a \cdot (\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g}) & \xrightarrow{\quad \gamma((\boldsymbol{f}, \boldsymbol{g}),(\boldsymbol{f}', \boldsymbol{g}')) \quad} & & & a \cdot (\boldsymbol{f}, \boldsymbol{g}) \cdot (\boldsymbol{f}', \boldsymbol{g}')
\end{array} \qquad (4)$$

(To see this, note that the diagram commutes on the nose when $(\boldsymbol{f}', \boldsymbol{g}')$ is the identity; thus there is a sequential J-term witnessing the two composites are isomorphic, and hence equal by 1-truncation.)

Now consider the following diagram:

$$\begin{array}{ccccccc}
a \cdot (\boldsymbol{f}'\boldsymbol{f}) & \xrightarrow{\gamma(\boldsymbol{f}, \boldsymbol{f}')} & a \cdot \boldsymbol{f} \cdot \boldsymbol{f}' & \xrightarrow{a\dagger(\boldsymbol{f}, \boldsymbol{g})\cdot \boldsymbol{f}'} & a \cdot (\boldsymbol{f}, \boldsymbol{g}) \cdot \boldsymbol{f}' & \xrightarrow{h \cdot \boldsymbol{f}'} & a' \cdot \boldsymbol{f}' \\
{\scriptstyle (a \cdot \boldsymbol{f})\dagger(\boldsymbol{f}', \boldsymbol{g}')} \downarrow & & {\scriptstyle (a \cdot (\boldsymbol{f}, \boldsymbol{g}))\dagger(\boldsymbol{f}', \boldsymbol{g}')} \downarrow & & & & \downarrow {\scriptstyle a'\dagger(\boldsymbol{f}', \boldsymbol{g}')} \\
a \cdot \boldsymbol{f} \cdot (\boldsymbol{f}', \boldsymbol{g}') & \xrightarrow[{(a\dagger(\boldsymbol{f}, \boldsymbol{g}))\cdot(\boldsymbol{f}', \boldsymbol{g}')}]{} & a \cdot (\boldsymbol{f}, \boldsymbol{g}) \cdot (\boldsymbol{f}', \boldsymbol{g}') & \xrightarrow[{h \cdot (\boldsymbol{f}', \boldsymbol{g}')}]{} & a' \cdot (\boldsymbol{f}', \boldsymbol{g}') & \xrightarrow{h'} & a''
\end{array}$$

The two squares commute by naturality of the operation $- \dagger (\boldsymbol{f}', \boldsymbol{g}')$ (which is established by the same standard argument again). Now one way around the diagram corresponds to the composite

$$\begin{aligned}
\tau_{\Gamma;\Delta;A}\big(\boldsymbol{f}', \boldsymbol{g}', h'\big) \circ \tau_{\Gamma;\Delta;A}\big(\boldsymbol{f}, \boldsymbol{g}, h\big) &= \big(\boldsymbol{f}', h' \circ a' \dagger (\boldsymbol{f}', \boldsymbol{g}')\big)\big(\boldsymbol{f}, h \circ a' \dagger (\boldsymbol{f}, \boldsymbol{g})\big) \\
&= \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ a \dagger (\boldsymbol{f}', \boldsymbol{g}') \circ ((h \circ a \dagger (\boldsymbol{f}, \boldsymbol{g})) \cdot \boldsymbol{f}') \circ \gamma(\boldsymbol{f}, \boldsymbol{f}')\big) \\
&= \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ a \dagger (\boldsymbol{f}', \boldsymbol{g}') \circ (h \cdot \boldsymbol{f}') \circ (a \dagger (\boldsymbol{f}, \boldsymbol{g}) \cdot \boldsymbol{f}') \circ \gamma(\boldsymbol{f}, \boldsymbol{f}')\big)
\end{aligned}$$

Moreover, the following calculation shows that the other way around the diagram corresponds to $\tau_{\Gamma;\Delta;A}\big((\boldsymbol{f}', \boldsymbol{g}', h') \circ (\boldsymbol{f}, \boldsymbol{g}, h)\big)$:

$$\begin{aligned}
\tau_{\Gamma;\Delta;A}\big((\boldsymbol{f}', \boldsymbol{g}', h') \circ (\boldsymbol{f}, \boldsymbol{g}, h)\big) &= \tau_{\Gamma;\Delta;A}\big(\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}')} \boldsymbol{g}), h' \circ_{(\boldsymbol{f}, \boldsymbol{g}),(\boldsymbol{f}', \boldsymbol{g}')} h\big) \\
&= \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ_{(\boldsymbol{f}, \boldsymbol{g}),(\boldsymbol{f}', \boldsymbol{g}')} h \circ a \dagger (\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g})\big) \\
&= \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ (h \cdot (\boldsymbol{f}', \boldsymbol{g}')) \circ \gamma((\boldsymbol{f}, \boldsymbol{g}),(\boldsymbol{f}', \boldsymbol{g}')) \circ a \dagger (\boldsymbol{f}'\boldsymbol{f}, \boldsymbol{g}' \circ_{\boldsymbol{f}, \boldsymbol{f}'} \boldsymbol{g})\big) \\
&= \big(\boldsymbol{f}'\boldsymbol{f}, h' \circ (h \cdot (\boldsymbol{f}', \boldsymbol{g}')) \circ (a \dagger (\boldsymbol{f}, \boldsymbol{g})) \cdot (\boldsymbol{f}', \boldsymbol{g}') \circ (a \cdot \boldsymbol{f}) \dagger (\boldsymbol{f}', \boldsymbol{g}') \circ \gamma(\boldsymbol{f}, \boldsymbol{f}')\big)
\end{aligned}$$

where the last step uses the commutativity of (4). This concludes the proof that τ is functorial.

We also observe:

Lemma 3.1. *There is a pullback diagram*

$$
\begin{array}{ccc}
|\Gamma, \Delta, x : A| & \xrightarrow{\ \tau_{\Gamma;\Delta;A}\ } & |\Gamma, x : A| \\
\downarrow & & \downarrow \\
|\Gamma, \Delta| & \longrightarrow & |\Gamma|.
\end{array}
\tag{5}
$$

Proof. This is straightforward: given a groupoid $\mathcal{H}$ and functors $L : \mathcal{H} \to |\Gamma, \Delta|$, $M : \mathcal{H} \to |\Gamma, x : A|$ making the outer diagram commute, we define $K : \mathcal{H} \to |\Gamma, \Delta, x : A|$ by $K(u) = (\boldsymbol{c}, \boldsymbol{d}, a)$, where $L(u) = (\boldsymbol{c}, \boldsymbol{d})$ and $M(u) = (\boldsymbol{c}, a)$. For an arrow $l : u \to v$ in $\mathcal{H}$, define

$$
K(l) = (\boldsymbol{f}, \boldsymbol{g}, h \circ (a \dagger (\boldsymbol{f}, \boldsymbol{g}))^{-1})
$$

where $L(l) = (\boldsymbol{f}, \boldsymbol{g})$ and $M(l) = (\boldsymbol{f}, h)$. $\qquad\square$

3.5. **Generalized form.** We shall have the need for a slightly more general form of weakening than the one described above, namely:

$$
\frac{\Gamma, \Theta \vdash \qquad \Delta \vdash}{\Gamma, \Delta, \Theta \vdash}
$$

In this situation, we wish to define a functor

$$
\tau_{\Gamma;\Delta;\Theta} : |\Gamma, \Delta, \Theta| \to |\Gamma, \Theta|
$$

which fits into a pullback square

$$
\begin{array}{ccc}
|\Gamma, \Delta, \Theta| & \xrightarrow{\ \tau_{\Gamma;\Delta;\Theta}\ } & |\Gamma, \Theta| \\
\downarrow & & \downarrow \\
|\Gamma, \Delta| & \longrightarrow & |\Gamma|.
\end{array}
\tag{6}
$$

This is done by induction on the length of Θ, where the base case has been addressed in the previous section. For the inductive case, we assume that we have defined $\tau_{\Gamma;\Delta;\Theta} : |\Gamma, \Delta, \Theta| \to |\Gamma, \Theta|$ (fitting into the appropriate pullback square) and now need to define $\tau_{\Gamma;\Delta;\Theta,A} : |\Gamma, \Delta, \Theta, x : A| \to |\Gamma, \Theta, x : A|$. This is done, *mutatis mutandis*, in much the same way as the simpler case in the previous section: on objects, the functor sends an object $(\boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, a)$ to $(\boldsymbol{c}, \boldsymbol{e}, a)$. For a morphism $(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{k}, h) : (\boldsymbol{c}, \boldsymbol{d}, \boldsymbol{e}, a) \to (\boldsymbol{c}', \boldsymbol{d}', \boldsymbol{e}', a')$, we note first that by inductive hypothesis we are given

$$
\boldsymbol{e} \dagger (\boldsymbol{f}, \boldsymbol{g}) : \boldsymbol{e} \cdot \boldsymbol{f} \to \boldsymbol{e} \cdot (\boldsymbol{f}, \boldsymbol{g}).
$$

Then there is a canonical term

$$
a \dagger (\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{k}) : a \cdot (\boldsymbol{f}, \boldsymbol{k} \cdot \boldsymbol{e} \dagger (\boldsymbol{f}, \boldsymbol{g})) \to a \cdot (\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{k})
$$

(which is defined by an appropriate J-term). Then the action of $\tau_{\Gamma;\Delta;\Theta,A} : |\Gamma, \Delta, \Theta, x : A| \to |\Gamma, \Theta, x : A|$ on the morphism $(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{k}, h)$ is defined to be $(\boldsymbol{f}, \boldsymbol{k} \cdot \boldsymbol{e} \dagger (\boldsymbol{f}, \boldsymbol{g}), h \cdot a \dagger (\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{k}))$, and we obtain a pullback square

$$
\begin{array}{ccc}
|\Gamma, \Delta, \Theta, x : A| & \longrightarrow & |\Gamma, \Theta, x : A| \\
\downarrow & & \downarrow \\
|\Gamma, \Delta, \Theta| & \longrightarrow & |\Gamma, \Theta|
\end{array}
$$

The result then follows from elementary properties of pullbacks. We note for future reference the following special case:

14

Lemma 3.2. *Let Γ and Δ be valid contexts, and let $\Gamma \vdash A$ be a type judgement. Then the weakening functors $|\Delta, \Gamma| \to |\Gamma|$ and $|\Delta, \Gamma, x : A| \to |\Gamma, x : A|$ fit into a pullback diagram*

$$
\begin{array}{ccc}
|\Delta, \Gamma, x : A| & \longrightarrow & |\Gamma, x : A| \\
\downarrow & & \downarrow \\
|\Delta, \Gamma| & \longrightarrow & |\Gamma|
\end{array}
$$

We remark that the substitution functors satisfy a coherence principle: any diagram built up out of these functors will automatically be commutative. On objects, this is virtually immediate from the definition of these functors, while on arrows it is a consequence of 1-truncation.

3.6. **Substitution.** We next introduce a family of functors which form the categorical counterpart of substitution. Consider a substitution instance

$$
\frac{\Gamma \ \vdash\ a : A \qquad \Gamma, x : A, \Delta \ \vdash}{\Gamma, \Delta[a/x] \ \vdash\ .}
$$

Given this data, we will define a functor

$$
\sigma_{\Gamma; a; \Delta} : |\Gamma, \Delta[a/x]| \to |\Gamma, x : A, \Delta|. \tag{7}
$$

To this end, we first recall from Section 3.3 that the judgement $\Gamma \ \vdash\ a : A$ gives a functor $|a| : |\Gamma| \to |\Gamma, x : A|$ which sends an object $\boldsymbol{c}$ to $(\boldsymbol{c}, a(\boldsymbol{c}))$ and which acts on morphisms by sending $\boldsymbol{h} : \boldsymbol{c} \to \boldsymbol{d}$ to $(\boldsymbol{h}, a|\boldsymbol{h})$, where $a|\boldsymbol{h} : a(\boldsymbol{c}) \cdot \boldsymbol{h} \to a(\boldsymbol{d})$.

Therefore to define the functor $\sigma_{\Gamma; a; \Delta}$, we set

$$
\sigma_{\Gamma; a; \Delta}(\boldsymbol{c}, \boldsymbol{d}) =_{\mathrm{def}} (\boldsymbol{c}, a(\boldsymbol{c}), \boldsymbol{d}).
$$

To define the action on morphisms, we proceed by induction on the length of the context, first assuming that $\Delta = (v_1 : B_1)$. Consider a morphism $(\boldsymbol{f}, g) : (\boldsymbol{c}, d_1) \to (\boldsymbol{c}', d_1')$. Of course the first two components of $\sigma_{\Gamma; a; \Delta}(\boldsymbol{f}, g)$ should be $\boldsymbol{f}$ and $a|\boldsymbol{f}$. The third component should thus be of the form $d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \to d_1'$.

Now d_1 is an object of $|B_1(\boldsymbol{c}, a(\boldsymbol{c}))|$, so that $d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f})$ is an object of $|B_1(\boldsymbol{c}', a(\boldsymbol{c}'))|$. Note first that there is a comparison isomorphism

$$
d_1 \ddagger \boldsymbol{f} : d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \to d_1 \cdot \boldsymbol{f}
$$

in $|B_1(\boldsymbol{c}', a(\boldsymbol{c}'))|$, arising by taking the parameterized sequential J-term

$$
\frac{\tilde{\Gamma}, u_1 : B_1(\boldsymbol{x}, a(\boldsymbol{x})) \ \vdash\ \underline{B_1(\boldsymbol{x}, a(\boldsymbol{x}))}(u_1 \cdot (\boldsymbol{z}, a|\boldsymbol{z}), u_1 \cdot \boldsymbol{z})}{\dfrac{\Gamma, u_1 : B_1(\boldsymbol{x}, a(\boldsymbol{x})) \ \vdash\ \mathrm{r}(u_1) : \underline{B_1(\boldsymbol{x}, a(\boldsymbol{x}))}(u_1, u_1)}{\dfrac{\boldsymbol{f} : \boldsymbol{c} \to \boldsymbol{c}'}{u_1 : B_1(\boldsymbol{c}, a(\boldsymbol{c})) \ \vdash\ \mathrm{J}^{\sigma}(\mathrm{r}(u_1), \boldsymbol{c}, \boldsymbol{c}', \boldsymbol{f}, u_1) : \underline{B_1(\boldsymbol{c}, a(\boldsymbol{c}))}(u_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}), u_1 \cdot \boldsymbol{f})}}}
$$

(where we schematically denote the variables in $\tilde{\Gamma}$ by $\boldsymbol{x}, \boldsymbol{y}$ and $\boldsymbol{z}$) and setting $u_1 =_{\mathrm{def}} d_1$.

Lemma 3.3. *The family of morphisms $- \ddagger \boldsymbol{f}$ is natural in its first argument.*

Proof. Given $g_1 : d_1 \to d_1'$, we note that there is a propositional equality in the naturality square

$$
\begin{array}{ccc}
d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) & \xrightarrow{\ g_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \ } & d_1' \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \\
{\scriptstyle d_1 \ddagger \boldsymbol{f}} \downarrow & & \downarrow {\scriptstyle d_1' \ddagger \boldsymbol{f}} \\
d_1 \cdot \boldsymbol{f} & \xrightarrow[\ g_1 \cdot \boldsymbol{f} \ ]{} & d_1' \cdot \boldsymbol{f}
\end{array}
$$

Thus by 1-truncation, the square commutes. $\qquad\square$

We now set $\sigma_{\Gamma;a;\Delta}(\boldsymbol{f}, g_1) = (\boldsymbol{f}, a|\boldsymbol{f}, g_1 \circ d_1 \ddagger \boldsymbol{f})$, where the last component is the composite

$$d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \xrightarrow{d_1 \ddagger \boldsymbol{f}} d_1 \cdot \boldsymbol{f} \xrightarrow{g_1} d_1'$$

Lemma 3.4. *As defined, $\sigma_{\Gamma;a;\Delta} : |\Gamma, \Delta[a/x]| \to |\Gamma, x : A, \Delta|$ is functorial.*

Proof. Consider, in addition to $(\boldsymbol{f}, g_1) : (\boldsymbol{c}, d_1) \to (\boldsymbol{c}', d_1')$, another morphism $(\boldsymbol{f}', g_1') : (\boldsymbol{c}', d_1') \to (\boldsymbol{c}'', d_1'')$, giving rise to a composite $(\boldsymbol{f}'\boldsymbol{f}, g_1' \circ_{\boldsymbol{f},\boldsymbol{f}'} g_1)$.
Consider the diagram

$$
\begin{array}{ccccc}
d_1 \cdot (\boldsymbol{f}, a|\boldsymbol{f}) \cdot (\boldsymbol{f}', a|\boldsymbol{f}') & \xrightarrow{d_1 \ddagger \boldsymbol{f} \cdot (\boldsymbol{f}', a|\boldsymbol{f}')} & d_1 \cdot \boldsymbol{f} \cdot (\boldsymbol{f}', a|\boldsymbol{f}') & \xrightarrow{g_1 \cdot (\boldsymbol{f}', a|\boldsymbol{f}')} & d_1' \cdot (\boldsymbol{f}', a|\boldsymbol{f}') \\
{\scriptstyle \gamma}\uparrow & & \downarrow{\scriptstyle (d_1 \cdot \boldsymbol{f}) \ddagger \boldsymbol{f}'} & & \downarrow{\scriptstyle d_1' \ddagger \boldsymbol{f}'} \\
d_1 \cdot (\boldsymbol{f}'\boldsymbol{f}, a|\boldsymbol{f}'\boldsymbol{f}) & \xrightarrow[d_1 \ddagger \boldsymbol{f}'\boldsymbol{f}]{} & d_1 \cdot \boldsymbol{f} \cdot \boldsymbol{f}' & \xrightarrow[g_1 \cdot \boldsymbol{f}']{} & d_1'' \cdot \boldsymbol{f}' \xrightarrow[g_1']{} d_1''
\end{array}
$$

The vertical left-hand map is the coherence isomorphism for the action of $|\Gamma, x : A \vdash \Delta|$. The right-hand square commutes by naturality of $\ddagger$. The left-hand square can also be seen to commute using the by now familiar argument: when $\boldsymbol{f}'$ is a reflexivity, the square commutes, and hence there is a propositional identity, and by 1-truncation, an identity between the two composites.

The composite starting with γ, going along the top and then down is equal to $\sigma_{\Gamma;a;\Delta}(\boldsymbol{f}', g_1') \circ \sigma_{\Gamma;a;\Delta}(\boldsymbol{f}, g_1)$, while the bottom composite is $\sigma_{\Gamma;a;\Delta}(\boldsymbol{f}'\boldsymbol{f}, g_1' \circ_{\boldsymbol{f}',\boldsymbol{f}} g_1)$. $\qquad\square$

Next, we use the same construction (but with extra parameters d_1, d_1', g_1) to get a comparison morphism

$$d_2 \ddagger (\boldsymbol{f}, g_1) : d_2 \cdot (\boldsymbol{f}, a|\boldsymbol{f}, g_1 \circ (d_1 \ddagger \boldsymbol{f})) \to d_2 \cdot (\boldsymbol{f}, g_1).$$

Thus we put $\sigma_{\Gamma;a;\Delta}(\boldsymbol{f}, g_1, g_2) = (\boldsymbol{f}, a|\boldsymbol{f}, g_1 \circ d_1 \ddagger \boldsymbol{f}, g_2 \circ d_2 \ddagger (\boldsymbol{f}, g_1))$. In general we have a comparison isomorphism

$$d_{m+1} \ddagger (\boldsymbol{f}, \boldsymbol{g}) : d_{m+1} \cdot (\boldsymbol{f}, a|\boldsymbol{f}, \boldsymbol{g} \circ (\boldsymbol{d} \ddagger (\boldsymbol{f}, \boldsymbol{g}))) \to d_{m+1} \cdot (\boldsymbol{f}, \boldsymbol{g})$$

(where now the list $\boldsymbol{g} = (g_1, \dots, g_m)$) and therefore we put

$$\sigma_{\Gamma;a;\Delta}(\boldsymbol{f}, \boldsymbol{g}) =_{\text{def}} (\boldsymbol{f}, a|\boldsymbol{f}, \boldsymbol{g} \circ (\boldsymbol{d} \ddagger (\boldsymbol{f}, \boldsymbol{g}))).$$

In much the same manner as in the base case we can now show that $- \ddagger (\boldsymbol{f}, \boldsymbol{g})$ is natural in its first argument, and that $\sigma_{\Gamma;a;\Delta}$, as defined, is functorial.

For future reference, we also collect the following fact about the substitution functors.

Lemma 3.5. *The functor $\sigma_{\Gamma;a;\Delta}$ fits into a pullback square*

$$
\begin{array}{ccc}
|\Gamma, \Delta[a/x]| & \xrightarrow{\sigma_{\Gamma;a;\Delta}} & |\Gamma, x : A, \Delta| \\
\downarrow & & \downarrow \\
|\Gamma| & \xrightarrow[|a|]{} & |\Gamma, x : A|
\end{array}
$$

Moreover, given an extended context $\Gamma, x : A, \Delta, \Delta'$, we have a pullback square

$$
\begin{array}{ccc}
|\Gamma, \Delta[a/x], \Delta'[a/x]| & \xrightarrow{\sigma_{\Gamma;a;\Delta,\Delta'}} & |\Gamma, x : A, \Delta, \Delta'| \\
\downarrow & & \downarrow \\
|\Gamma, \Delta[a/x]| & \xrightarrow[\sigma_{\Gamma;a;\Delta}]{} & |\Gamma, x : A, \Delta|
\end{array}
$$

Proof. It is clear that the square commutes. Given a groupoid $\mathcal{H}$ and functors $L : \mathcal{H} \to |\Gamma|$, $M : \mathcal{H} \to |\Gamma, x : A, \Delta|$ making the outer diagram commute, define $K : \mathcal{H} \to |\Gamma, \Delta[a/x]|$ by $K(u) = (\boldsymbol{c}, \boldsymbol{d})$, where $L(u) = \boldsymbol{c}$ and $M(u) = (\boldsymbol{c}, a(\boldsymbol{c}), \boldsymbol{d})$. For an arrow $l : u \to v$ in $\mathcal{H}$ define

$$K(l) =_{\mathrm{def}} \big(\boldsymbol{f}, a|\boldsymbol{f}, \boldsymbol{g} \circ (\boldsymbol{d} \ddagger (\boldsymbol{f}, \boldsymbol{g}))\big),$$

where $L(u) = \boldsymbol{f}$, and $K(u) = (\boldsymbol{f}, a|\boldsymbol{f}, \boldsymbol{g})$.

The second claim is immediate from elementary properties of pullback squares. $\qquad\square$

3.7. Syntactical groupoid CwF.
Our last aim for this section is to organize the groupoids and pseudo-functors associated to contexts and types into a genuine model of the type theory.

Write **Cont** for the category of contexts and context morphisms. This is the underlying category of a CwF, the syntactic model of the 1-truncated type theory (see [5]). We shall use the constructions from the previous section to define a functor $|-| : \textbf{Cont} \to \textbf{Gpd}$.

To a context $\Gamma = (x_1 : T_1, \ldots, x_k : T_k)$ we associate the syntactic groupoid $|\Gamma|$. Next suppose that we are given another context Δ and a context morphism $\boldsymbol{m} : \Delta \to \Gamma$, i.e., we have a sequence of derivable term judgements

$$
\begin{aligned}
\Delta &\vdash m_1 : T_1 \\
\Delta &\vdash m_2 : T_2[m_1/x_1] \\
&\vdots \\
\Delta &\vdash m_k : T_n[m_1/x_1, \ldots, m_{k-1}/x_{k-1}].
\end{aligned}
$$

Then we wish to construct a functor

$$|\boldsymbol{m}| : |\Delta| \to |\Gamma|.$$

To this end, first note that the terms m_i induce sections $|m_i|$ of the appropriate projections. These fit together in the following manner (illustrated for the case $k = 3$):

Here the squares are pullbacks (cf. Lemma 3.5). Generally, given $\boldsymbol{m} : \Delta \to \Gamma$, we denote by $|\overline{\boldsymbol{m}}|$ the composite

17

$$|\Delta|$$
$$\downarrow |m_k|$$
$$|\Delta, x_k : T_k[m_1, \ldots, m_{k-1}]|$$
$$\downarrow \sigma_{\Delta; m_{k-1}; T_k}$$
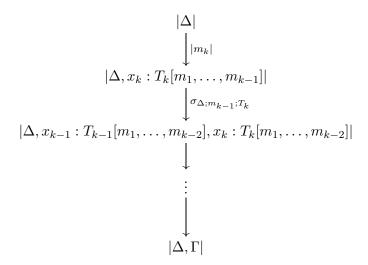$$|\Delta, x_{k-1} : T_{k-1}[m_1, \ldots, m_{k-2}], x_k : T_k[m_1, \ldots, m_{k-2}]|$$
$$\downarrow$$
$$\vdots$$
$$\downarrow$$
$$|\Delta, \Gamma|$$

Then we define the functor $|\boldsymbol{m}|$ associated to the context morphism $\boldsymbol{m}$ to be the composite

$$|\Delta| \xrightarrow{\ |\overline{\boldsymbol{m}}|\ } |\Delta, \Gamma| \xrightarrow{\ \tau_{\Delta;\Gamma}\ } |\Gamma|$$

where $\tau_{\Delta;\Gamma}$ is the weakening functor described in section 3.5.

Just as a single term $\Delta \vdash a : T$ induces substitution functors $\sigma_{\Delta;a;\Theta} : |\Delta, \Theta[a/x]| \to |\Delta, x : A, \Theta|$, a context morphism $\boldsymbol{m} : \Delta \to \Gamma$ also induces substitution functors $\sigma_{\Delta;\boldsymbol{m};\Theta} : |\Delta, \Theta[\boldsymbol{m}/\boldsymbol{x}]| \to |\Delta, \Gamma, \Theta|$ fitting into a pullback square

$$
\begin{array}{ccc}
|\Delta, \Theta[\boldsymbol{m}]| & \xrightarrow{\ \sigma_{\Delta;\boldsymbol{m};\Theta}\ } & |\Delta, \Gamma, \Theta| \\
\downarrow & & \downarrow \\
|\Delta| & \xrightarrow[\ \overline{\boldsymbol{m}}\ ]{} & |\Delta, \Gamma|
\end{array}
$$

These functors can also be defined explicitly, in a manner very similar to that of the usual substitution functors; the details are left to the reader.

Proposition 3.6. *The above definitions give a functor* $|-| : \mathbf{Cont} \to \mathbf{Gpd}$.

Proof. Given two composable context morphisms $\boldsymbol{n} : \Theta \to \Delta$ and $\boldsymbol{m} : \Delta \to \Gamma$, we have, on the one hand, the composite functor

$$|\Theta| \xrightarrow{\ |\overline{\boldsymbol{m}}|\ } |\Theta, \Delta| \xrightarrow{\ \tau_{\Theta;\Delta}\ } |\Delta| \xrightarrow{\ |\overline{\boldsymbol{n}}|\ } |\Delta, \Gamma| \xrightarrow{\ \tau_{\Delta;\Gamma}\ } |\Gamma|$$

On the other hand, we have the functor associated to the composite context morphism $\boldsymbol{m}[\boldsymbol{n}/\boldsymbol{x}]$, which has the form

$$|\Theta| \xrightarrow{\ |\boldsymbol{m}[\boldsymbol{n}/\boldsymbol{x}]|\ } |\Theta, \Gamma| \xrightarrow{\ \tau_{\Theta;\Gamma}\ } |\Gamma|.$$

In order to show that these are equal, consider the diagram

$$
\begin{array}{ccccc}
|\Theta| & \xrightarrow{\;|\overline{\boldsymbol{n}}|\;} & |\Theta, \Delta| & \xrightarrow{\;\tau_{\Theta;\Delta}\;} & |\Delta| \\
{\scriptstyle |\overline{\boldsymbol{m}[\boldsymbol{n}/\boldsymbol{x}]}|} \downarrow & & \downarrow {\scriptstyle |\overline{\boldsymbol{m}}|} & & \downarrow {\scriptstyle |\overline{\boldsymbol{m}}|} \\
|\Theta, \Gamma| & \xrightarrow{\;\sigma_{\Theta;\boldsymbol{n};\Gamma}\;} & |\Theta, \Delta, \Gamma| & \xrightarrow{\;\tau_{\Theta;\Delta,\Gamma}\;} & |\Delta, \Gamma| \xrightarrow{\;\tau_{\Delta;\Gamma}\;} |\Gamma| \\
\downarrow & & \downarrow & & \downarrow \\
|\Theta| & \xrightarrow{\;|\overline{\boldsymbol{n}}|\;} & |\Theta, \Delta| & \xrightarrow{\;\tau_{\Theta;\Delta}\;} & |\Delta|
\end{array}
$$

(We have overloaded the notation $|\boldsymbol{m}|$; the middle vertical arrow is the weakened version.) All squares are pullbacks by general properties of substitution and weakening functors. Moreover, the vertical composites are identities. The result now follows from the commutativity of

$$
\begin{array}{ccc}
|\Theta, \Gamma| & \xrightarrow{\;\sigma_{\Theta;\boldsymbol{n};\Gamma}\;} & |\Theta, \Delta, \Gamma| \\
{\scriptstyle \tau_{\Theta;\Gamma}} \downarrow & & \downarrow {\scriptstyle \tau_{\Theta;\Delta,\Gamma}} \\
|\Gamma| & \xleftarrow{\;\tau_{\Delta;\Gamma}\;} & |\Delta, \Gamma|
\end{array}
$$

which is a consequence of 1-truncation. $\qquad\square$

We next use the functor $|-| : \mathbf{Cont} \to \mathbf{Gpd}$ to transfer the additional CwF-structure on the category $\mathbf{Cont}$ of contexts to the category of groupoids; more precisely, we shall define a CwF-structure on the image of the functor $|-|$. We shall write $\mathbf{SyntGpd}$ for the image of the functor $|-|$. Its objects and arrows will be referred to as *syntactic groupoids* and *syntactic functors*, respectively.

Theorem 3.7. *The category* $\mathbf{SyntGpd}$ *is the underlying category of a CwF, and the functor* $|-| : \mathbf{Cont} \to \mathbf{SyntGpd} \hookrightarrow \mathbf{Gpd}$ *induces an isomorphism of CwFs.*

Since the functor $|-|$ is faithful and injective on objects, it is immediate that its image can be made into a CwF which is isomorphic to $\mathbf{Cont}$. However, as we shall now explain, more can be said, since the CwF structure on the image can be given directly through more intuitive constructions.

Given a syntactic context $|\Gamma|$, the collection of *types* over $|\Gamma|$ is that of the syntactic model, i.e. it is the set of types A in context Γ. As described in section 2, such a type gives rise to a pseudofunctor $|\Gamma| \to \mathbf{Gpd}$, whose Grothendieck construction defines the syntactic groupoid $|\Gamma, x : A|$, with first projection $p : |\Gamma, x : A| \to |\Gamma|$. Then by a *syntactic section* of p we mean a section of p which is actually a context morphism. It is easily seen that such a section corresponds uniquely to the interpretation of a definitional equality class of terms $\Gamma \vdash t : A$. We thus may define a *term* over $\Gamma \vdash A$ to be such a syntactic section.

For later use we also record the following easy lemma, the proof of which is easily extracted from the calculations done so far:

Lemma 3.8. *Given a context morphism $\boldsymbol{m} : \Delta \to \Gamma$ and a term $\Gamma \vdash t : A$ we have a diagram*

$$
\begin{array}{ccc}
|\Delta| & \xrightarrow{\ \ \boldsymbol{m}\ \ } & |\Gamma| \\
{\scriptstyle t[\boldsymbol{m}/\boldsymbol{x}]}\Big\downarrow & & \Big\downarrow{\scriptstyle |t|} \\
|\Delta, x : A[\boldsymbol{m}/\boldsymbol{x}]| & \longrightarrow & |\Gamma, x : A| \\
\Big\downarrow & & \Big\downarrow \\
|\Delta| & \xrightarrow[\ \ \boldsymbol{m}\ \ ]{} & |\Gamma|
\end{array}
$$

in which the squares are pullbacks and the vertical composites are the identity.

4. The model

Throughout this section, we work with a fixed theory of the form $\mathbb{T}_1[G]$, where G is a graph. In the previous section, we have organized syntactic groupoids into a model in the form of a category with families. In this section, we extend this model by adding a notion of realizer. We first explain the general form the resulting CwF will take, and then investigate the semantic type formers one by one.

4.1. **The Realizability CwF.** So far we have constructed a category with families **SyntGpd** of syntactic groupoids. This is a sound and complete model of the 1-truncated type theory, but for us it is only an intermediate step, as we shall now glue this model with a notion of realizability. This model shall again take the form of a CwF. For clarity, we stress that the CwF **SyntGpd**, even though the underlying category is a subcategory of the category of groupoids, has a CwF structure which does not agree with the Hofmann-Streicher model on the category of groupoids.

The underlying category of the realizability CwF will be called $\mathcal{V}$, and it is defined as follows:

Objects: cloven fibrations $\mathbb{C} \to |\Gamma|$, where Γ is a context and $\mathbb{C}$ is a groupoid.
Morphisms: are pairs $(F, |\boldsymbol{m}|)$ forming commutative squares

$$
\begin{array}{ccc}
\mathbb{C} & \longrightarrow & |\Gamma| \\
{\scriptstyle F}\Big\downarrow & & \Big\downarrow{\scriptstyle |\boldsymbol{m}|} \\
\mathbb{D} & \longrightarrow & |\Delta|
\end{array}
$$

Thus, $\mathcal{V}$ is the full subcategory on the gluing of $|-| : \mathbf{SyntGpd} \to \mathbf{Gpd}$ determined by the cloven fibrations. In what follows, we shall often denote a typical object of $\mathcal{V}$ by $\pi_\Gamma : [\![\Gamma]\!] \to |\Gamma|$, and a typical morphism from π_Γ to π_Δ by $([\![\boldsymbol{m}]\!], |\boldsymbol{m}|)$. Of course, the domain of an object in $\mathcal{V}$ is not necessarily determined in any way by its codomain, but the objects which arise by successive applications of comprehension will be determined completely by their syntactic part. Anticipating the format of the interpretation even more, we will write the objects of $[\![\Gamma]\!]$ as $(\boldsymbol{c}, \boldsymbol{\gamma})$, where $\boldsymbol{c}$ is an object of the groupoid $|\Gamma|$ and $\boldsymbol{\gamma}$ is an object in the fiber of π_Γ over $\boldsymbol{c}$.

We note that $\mathcal{V}$ has a terminal object, namely the identity morphism on the terminal groupoid, written $[\![()]\!] \to |()|$.

The CwF we now define will be called $\mathbf{Real}(\mathcal{R})$, where $\mathcal{R} : |G| \to \mathbf{Set}$ is a basic notion of realizer (see section 2 for a discussion).

  Underlying category: the category $\mathcal{V}$;
  Semantic Types: Given a semantic context $\pi_\Gamma : [\![\Gamma]\!] \to |\Gamma|$, a *semantic type* over π_Γ consists of the following data:
    • A type $\Gamma \vdash A$ in context Γ.

- A functor $[\![\Gamma \vdash A]\!] : \|\Gamma, x : A\| \to \mathbf{Set}$, where $\|\Gamma, x : A\|$ is the groupoid arising in the following pullback:

$$
\begin{array}{ccc}
\|\Gamma, x : A\| & \longrightarrow & |\Gamma, x : A| \\
\downarrow & & \downarrow \\
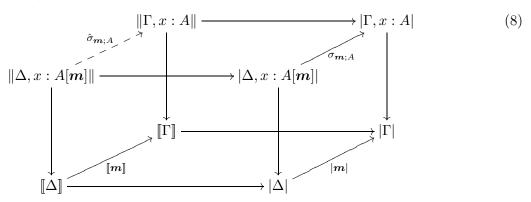[\![\Gamma]\!] & \xrightarrow{\ \pi_\Gamma\ } & |\Gamma|
\end{array}
$$

We denote such a type by $(\Gamma, x : A, \Vdash)$, where $\Vdash$ is the functor $\|\Gamma, x : A\| \to \mathbf{Set}$. Instead of $\tau \in \Vdash(\boldsymbol{c}, \boldsymbol{\gamma}, a)$ we write

$$
\tau \Vdash_{(\boldsymbol{c}, \boldsymbol{\gamma})} a : A(\boldsymbol{c})
$$

or, when no confusion will result, simply $\tau \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c})$.

Comprehension: Given a semantic type $(\Gamma, x : A, \Vdash)$, its comprehension consists of the fibration $[\![\Gamma, x : A]\!] \to |\Gamma, x : A|$, where the domain is the Grothendieck construction of the functor $\Vdash$. Then the fibre of the projection $[\![\Gamma, x : A]\!] \to [\![\Gamma]\!]$ over an object $(\boldsymbol{c}, \boldsymbol{\gamma})$ consists of objects of the form $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha)$ where $(\boldsymbol{c}, a)$ is an object of $|\Gamma, x : A|$ and $\alpha \Vdash_{(\boldsymbol{c}, \boldsymbol{\gamma})} a : A$.

Type Substitution: Given a semantic context morphism $([\![\boldsymbol{m}]\!], |\boldsymbol{m}|) : (\pi_\Delta : [\![\Delta]\!] \to |\Delta|) \to (\pi_\Gamma : [\![\Gamma]\!] \to |\Gamma|)$ and given a semantic type $(\Gamma, x : A, \Vdash)$ as above, we consider the diagram

$$
\tag{8}
$$

in which the bottom square commutes because $([\![\boldsymbol{m}]\!], |\boldsymbol{m}|)$ is a morphism, the right-hand square is a pullback by virtue of the properties of the substitution functors, and the front and back squares are pullbacks by definition. Thus there is a unique mediating arrow $\hat{\sigma}_{\boldsymbol{m};A} : \|\Delta, x : A[\boldsymbol{m}]\| \to \|\Gamma, x : A\|$ making the left hand square a pullback. We then define the composite

$$
\|\Delta, x : A[\boldsymbol{m}]\| \xrightarrow{\hat{\tau}_{\boldsymbol{m};A}} \|\Gamma, x : A\| \xrightarrow{\ \Vdash\ } \mathbf{Set}
$$

to be the realizer functor associate to the reindexed type. Note that as a consequence, we have a pullback square

$$
\begin{array}{ccc}
[\![\Delta, x : A[\boldsymbol{m}]]\!] & \xrightarrow{\ \overline{\tau}_{\boldsymbol{m};A}\ } & [\![\Gamma, x : A]\!] \\
\downarrow & & \downarrow \\
[\![\Delta]\!] & \xrightarrow{\ [\![\boldsymbol{m}]\!]\ } & [\![\Gamma]\!]
\end{array}
\tag{9}
$$

The functor $\overline{\tau}_{\boldsymbol{m};A}$ sends an object $(\boldsymbol{d}, \boldsymbol{\delta}, a, \alpha)$ to $(\boldsymbol{m}[\boldsymbol{d}/\boldsymbol{x}], \boldsymbol{m}[\boldsymbol{\delta}], a, \alpha)$, where $[\![m]\!](\boldsymbol{d}, \boldsymbol{\delta}) = (\boldsymbol{m}[\boldsymbol{d}/\boldsymbol{x}], \boldsymbol{m}[\boldsymbol{\delta}])$.

Semantic Terms: Given a type $(\Gamma, x : A, \Vdash)$ as above, we have the induced projection fibration $[\![\Gamma, x : A]\!] \to [\![\Gamma]\!]$; then a *semantic term* is a section $[\![s]\!]$ of this fibration which *descends*

to a syntactic section, in the sense that there is a syntactic section $|s| : |\Gamma| \to |\Gamma, x : A|$ making the following diagram commute:

$$
\begin{array}{ccccc}
[\![\Gamma, x : A]\!] & \longrightarrow & \|\Gamma, x : A\| & \longrightarrow & |\Gamma, x : A| \\
 & \nwarrow & & & \uparrow {\scriptstyle |s|} \\
 & {\scriptstyle [\![s]\!]} & & & \\
 & & [\![\Gamma]\!] & \xrightarrow{\ \pi_\Gamma\ } & |\Gamma|
\end{array}
$$

I.e., a term is simply given by a section of the corresponding display map in the category $\mathcal{V}$. We will write

$$[\![t]\!](\boldsymbol{c}, \boldsymbol{\gamma}) = (\boldsymbol{c}, \boldsymbol{\gamma}, t(\boldsymbol{c}), t[\boldsymbol{\gamma}])$$

to denote the action of such a section.

Term Substitution: Given a semantic term $([\![s]\!], |s|)$ as above, and given a context morphism $([\![\boldsymbol{m}]\!], |\boldsymbol{m}|)$ from $(\pi_\Delta : [\![\Delta]\!] \to |\Delta|)$ to $(\pi_\Gamma : [\![\Gamma]\!] \to |\Gamma|)$, note first that since the diagram (9) is a pullback, the section $[\![s]\!]$ induces a section of $[\![\Delta, x : A[\boldsymbol{m}]]\!] \to [\![\Delta]\!]$. This is easily seen to descend to the induced section of $|\Delta, x : A[\boldsymbol{m}]| \to |\Delta|$.

We first note that while in the above formulation pullbacks are used to define substitution for types and for terms, these are pullbacks of cloven fibrations, so there is no concern about coherence. We have:

Proposition 4.1. *Under the above definitions,* **Real**$(\mathcal{R})$ *is a category with families.*

The generating type G is interpreted in **Real**$(\mathcal{R})$ using the notion of realizer $\mathcal{R}$, as in

$$
\begin{array}{ccc}
[\![x : G]\!] = \int \mathcal{R} & & \\
\downarrow & & \\
\|x : G\| & \xrightarrow{\ =\ } & |x : G| \\
\downarrow & & \downarrow \\
[\![()]\!] & \xrightarrow{\ =\ } & |()|.
\end{array}
$$

For future reference, we also record the following immediate consequence of the definition, which describes the effect of term substitution in terms of realizers:

Lemma 4.2. *For a context morphism* $[\![\boldsymbol{m}]\!] : [\![\Delta]\!] \to [\![\Gamma]\!]$, *a type* $\Gamma \vdash A$ *and an object* $(\boldsymbol{d}, \boldsymbol{\delta})$ *of* $[\![\Delta]\!]$, *we have*

$$\tau \Vdash_{\boldsymbol{\delta}} a : A[\boldsymbol{m}] \ \text{iff}\ \tau \Vdash_{\boldsymbol{m}[\boldsymbol{\delta}]} a : A[\boldsymbol{m}]$$

The remainder of this section is dedicated to the proof of the main result:

Theorem 4.3. *Let G be a graph, and let $\mathcal{R} : |G| \to \mathbf{Set}$ be a notion of realizer for which basic terms have realizers, stable under reindexing. Then the CwF* **Real**$(\mathcal{R})$ *is a sound and complete model of* $\mathbb{T}_1[G]$.

Of course, completeness is a trivial consequence of the fact that the model contains a full and faithful copy of the syntactic model.

4.2. Semantic product types. Consider a context $[\![\Gamma]\!]$, a type A over this context with comprehension $[\![\Gamma, x : A]\!]$ and furthermore a type B over this comprehension, giving a further comprehension $[\![\Gamma, x : A, y : B(x)]\!]$. Thus we have a diagram

$$
\begin{array}{ccccc}
[\![\Gamma, x : A, y : B(x)]\!] & \longrightarrow & [\![\Gamma, x : A]\!] & \longrightarrow & [\![\Gamma]\!] \\
\downarrow & & \downarrow & & \downarrow \\
|\Gamma, x : A, y : B(x)| & \longrightarrow & |\Gamma, x : A| & \longrightarrow & |\Gamma|.
\end{array}
$$

We need to define the semantic dependent product type in context $[\![\Gamma]\!]$. This we take to consists first of the syntactic type $\Gamma \vdash \prod_{x:A} B(x)$, giving rise to the pullback diagram

$$
\begin{array}{ccc}
\|\Gamma, v : \prod_{x:A} B(x)\| & \longrightarrow & |\Gamma, v : \prod_{x:A} B(x)| \\
\downarrow & & \downarrow \\
[\![\Gamma]\!] & \longrightarrow & |\Gamma|.
\end{array}
$$

Second, we must specify the realizability presheaf

$$
\|\Gamma, v : \prod_{x:A} B(x)\| \to \mathbf{Set}.
$$

To this end, consider an object $(\boldsymbol{c}, \boldsymbol{\gamma})$ of $[\![\Gamma]\!]$; we have associated groupoids

$$
[\![\Gamma, x : A]\!]_{(\boldsymbol{c},\boldsymbol{\gamma})} \text{ and } [\![\Gamma, x : A, y : B(x)]\!]_{(\boldsymbol{c},\boldsymbol{\gamma})}
$$

obtained as the fibres of $[\![\Gamma, x : A]\!] \to [\![\Gamma]\!]$ and $[\![\Gamma, x : A, y : B(x)]\!] \to [\![\Gamma]\!]$ over $(\boldsymbol{c}, \boldsymbol{\gamma})$. Given an object $(\boldsymbol{c}, \boldsymbol{\gamma}, f)$ of $\|\Gamma, v : \prod_{x:A} B(x)\|$, we define

$$
\phi \Vdash_{\boldsymbol{\gamma}} f : \prod_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)
$$

when ϕ is a section

$$
[\![\Gamma, x : A]\!]_{(\boldsymbol{c},\boldsymbol{\gamma})} \xrightarrow{\phi} [\![\Gamma, x : A, y : B(x)]\!]_{(\boldsymbol{c},\boldsymbol{\gamma})}
$$

$$
\xcancel{=} \searrow \quad \swarrow
$$

$$
[\![\Gamma, x : A]\!]_{(\boldsymbol{c},\boldsymbol{\gamma})}
$$

of the canonical projection satisfying the conditions that

$$
\phi(a, \alpha) = \big(a, \alpha, \mathtt{app}(f, a), \phi\{\alpha\}\big), \text{ and}
$$
$$
\phi(m) = \big(m, \mathtt{app}(f, x)|(1_{\boldsymbol{c}}, m)\big),
$$

for $\alpha \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c})$ and $m : a \to a'$ in $|A(\boldsymbol{c})|$. Thus, ϕ sends realizers α of a to realizers $\phi\{\alpha\}$ of $\mathtt{app}(f, a)$ in a functorial way. In particular, ϕ must satisfy the condition that

$$
\phi\{\alpha\} \cdot \big(1_{\boldsymbol{c}}, m, \mathtt{app}(f, x)|(1_{\boldsymbol{c}}, m)\big) = \phi\{\alpha \cdot (1_{\boldsymbol{c}}, m)\} \tag{10}
$$

for every arrow $m : a \to a'$ in $|A(\boldsymbol{c})|$ and $\alpha \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c})$.

23

4.2.1. *Reindexing.* We must also define a functorial action on these realizers. To this end, suppose we are given, in addition to ϕ as above, a morphism $(\boldsymbol{h}, m) : (\boldsymbol{c}, \boldsymbol{\gamma}, f) \to (\boldsymbol{c}', \boldsymbol{\gamma}', f')$ in $\|\Gamma, v : \prod_{x:A} B(x)\|$. We must then define a realizer

$$\phi \cdot (\boldsymbol{h}, m) \Vdash_{\boldsymbol{\gamma}'} f' : \prod_{x:A(\boldsymbol{c}')} B(\boldsymbol{c}', x).$$

First, if $\alpha \Vdash_{\boldsymbol{\gamma}'} a : A(\boldsymbol{c}')$, then we have, by the functorial action on the realizer α, that $\alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}}) \Vdash_{\boldsymbol{\gamma}} a \cdot \boldsymbol{h}^{-1} : A(\boldsymbol{c})$. But then

$$\phi\{\alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})\} \Vdash_{\boldsymbol{\gamma}, \alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})} \mathtt{app}(f, a \cdot \boldsymbol{h}^{-1}) : B(\boldsymbol{c}, a \cdot \boldsymbol{h}^{-1}).$$

Now, we observe that the diagram

commutes, where we denote the coherence isomorphism $(a \cdot \boldsymbol{h}^{-1}) \cdot \boldsymbol{h} \to a$ by $\iota(\boldsymbol{h})_a$. It follows from this that there is a morphism

$$(\boldsymbol{h}, \iota(\boldsymbol{h})_a) : \big(\boldsymbol{c}, \boldsymbol{\gamma}, a \cdot \boldsymbol{h}^{-1}, \alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})\big) \to \big(\boldsymbol{c}', \boldsymbol{\gamma}', a, \alpha\big)$$

in $[\![\Gamma, x : A]\!]$. Moreover, in $\|\Gamma, x : A, y : B(x)\|$ we have an arrow

$$\zeta : \Big(\boldsymbol{c}, \boldsymbol{\gamma}, a \cdot \boldsymbol{h}^{-1}, \alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}}), \mathtt{app}(f, a \cdot \boldsymbol{h}^{-1})\Big) \to \big(\boldsymbol{c}', \boldsymbol{\gamma}', a, \alpha, \mathtt{app}(f', a)\big) \qquad (11)$$

given by

$$\zeta =_{\mathrm{def}} (\boldsymbol{h}, \iota(\boldsymbol{h})_a, \mathtt{J}(\mathtt{r}(\mathtt{app}(v, a)), f \cdot \boldsymbol{h}, f', m) \circ \mathtt{J}^{\sigma}(\mathtt{r}(\mathtt{app}(u, x)), \boldsymbol{c}, \boldsymbol{c}', \boldsymbol{h}, a, f))$$

where $\mathtt{J}^{\sigma}(\mathtt{r}(\mathtt{app}(u, x)), \boldsymbol{c}, \boldsymbol{c}', \boldsymbol{h}, a, f)$ is the term

$$\frac{\tilde{\Gamma}, x : A(\boldsymbol{w}), u : \prod_{x:A(\boldsymbol{v})} B(\boldsymbol{v}, x) \vdash \underline{B(\boldsymbol{w}, x)}\big(\mathtt{app}(u, x \cdot \boldsymbol{z}^{-1}) \cdot (\boldsymbol{z}, \iota(\boldsymbol{z})_x), \mathtt{app}(u \cdot \boldsymbol{z}, x)\big) \qquad \Gamma, x : A(\boldsymbol{v}), u : \prod_{x:A(\boldsymbol{v})} B(\boldsymbol{v}, x) \vdash \mathtt{r}(\mathtt{app}(u, x)) : \underline{B(\boldsymbol{v}, x)}\big(\mathtt{app}(u, x), \mathtt{app}(u, x)\big)}{\mathtt{J}^{\sigma}(\mathtt{r}(\mathtt{app}(u, x)), \boldsymbol{c}, \boldsymbol{c}', \boldsymbol{h}, a, f) : \underline{B(\boldsymbol{c}', a)}\big(\mathtt{app}(f, a \cdot \boldsymbol{h}^{-1}) \cdot (\boldsymbol{h}, \iota(\boldsymbol{h})_a), \mathtt{app}(f \cdot \boldsymbol{h}, a)\big)}$$

and $\mathtt{J}(\mathtt{r}(\mathtt{app}(v, a)), f \cdot \boldsymbol{h}, f', m)$ is the term

$$\frac{v, w : \prod_{x:A(\boldsymbol{c}')} B(\boldsymbol{c}', x), z : \prod_{x:A(\boldsymbol{c}')} B(\boldsymbol{c}', x)(v, w) \vdash \underline{B(\boldsymbol{c}', a)}(\mathtt{app}(v, a), \mathtt{app}(w, a)) \qquad v : \prod_{x:A(\boldsymbol{c}')} B(\boldsymbol{c}', x) \vdash \mathtt{r}(\mathtt{app}(v, a)) : \underline{B(\boldsymbol{c}', a)}(\mathtt{app}(v, a), \mathtt{app}(v, a))}{\mathtt{J}(\mathtt{r}(\mathtt{app}(v, a)), f \cdot \boldsymbol{h}, f', m) : \underline{B(\boldsymbol{c}', a)}(\mathtt{app}(f \cdot \boldsymbol{h}, a), \mathtt{app}(f', a))}$$

Finally, we then have that

$$\phi\{\alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})\} \cdot \zeta \Vdash_{\boldsymbol{c}', \boldsymbol{\gamma}', a, \alpha} \mathtt{app}(f', a) : B(\boldsymbol{c}', a).$$

As such, we define $\phi \cdot (\boldsymbol{h}, m)$ by

$$\big(\phi \cdot (\boldsymbol{h}, m)\big)(a, \alpha) =_{\mathrm{def}} \big(a, \alpha, \mathtt{app}(f', a), \phi\{\alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})\} \cdot \zeta\big)$$

We leave to the reader the tedious but straightforward verification that this is indeed functorial. Moreover, the construction of the dependent product types is stable under substitution: given a context morphism $([\![\boldsymbol{m}]\!], |\boldsymbol{m}|) : ([\![\Delta]\!] \to |\Delta) \to ([\![\Gamma]\!] \to |\Gamma|)$, we must verify that the reindexing of

24

$[\![\Gamma, v : \prod_{x:A} B(x)]\!]$ along $[\![m]\!]$ agrees with the dependent product of the reindexed $[\![\Delta, x : A[m], y : B[m]]\!]$. It suffices to show that the diagram

$$\|\Delta, v : \textstyle\prod_{A[m]} B[m]\| \longrightarrow \|\Gamma, v : \prod_A B\|$$

$$\Vdash \searrow \qquad \downarrow \Vdash$$

$$\mathbf{Set}$$

commutes. But this is an immediate consequence of the fact that the diagram

$$[\![\Delta, x : A[m], y : B[m]]\!] \longrightarrow [\![\Gamma, x : A, y : B]\!]$$

$$\downarrow \qquad\qquad\qquad \downarrow$$

$$[\![\Delta, x : A[m]]\!] \longrightarrow [\![\Gamma, x : A]\!]$$

is a pullback, as this guarantees, for each object $(\boldsymbol{d}, \boldsymbol{\delta})$ of $[\![\Delta]\!]$, a bijective correspondence between sections of $[\![\Delta, x : A[m], y : B[m]]\!]_{(\boldsymbol{d},\boldsymbol{\delta})} \to [\![\Delta, x : A[m]]\!]_{(\boldsymbol{d},\boldsymbol{\delta})}$ and sections of $[\![\Gamma, x : A, y : B]\!]_{(\boldsymbol{m}(\boldsymbol{d}), \boldsymbol{m}[\boldsymbol{\delta}])} \to [\![\Gamma, x : A]\!]_{(\boldsymbol{m}(\boldsymbol{d}), \boldsymbol{m}[\boldsymbol{\delta}])}$.

Proposition 4.4. *The CwF* $\mathbf{Real}(\mathcal{R})$ *supports dependent products.*

Proof. We have detailed the semantic type formation; what remains is to construct the appropriate abstraction and application terms and show that these have the requisite properties.

First, consider a semantic term s of type $([\![\Gamma, x : A]\!], \Gamma, x : A \vdash B(x))$. Then $s = ([\![s]\!], |s|)$, where $[\![s]\!] : [\![\Gamma, x : A]\!] \to [\![\Gamma, x : A, y : B(x)]\!]$ is the section of the appropriate projection, and $|s|$ is the corresponding syntactic section. We must define a term of dependent product type, i.e. a section of the projection

$$[\![\Gamma, \prod_{x:A} B(x)]\!] \to [\![\Gamma]\!].$$

Given $(\boldsymbol{c}, \boldsymbol{\gamma})$ an object of $[\![\Gamma]\!]$ and $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha)$ an object of $[\![\Gamma, x : A]\!]$ we need to define $\lambda_{x:A} s[\boldsymbol{\gamma}]$. In particular, we require a realizer

$$\big(\lambda_{x:A} s[\boldsymbol{\gamma}]\big)\{\alpha\} \Vdash_{\boldsymbol{\gamma}, \alpha} \mathtt{app}(\lambda_{x:A} s(\boldsymbol{c}), a) : B(\boldsymbol{c}, a).$$

As such, we define

$$\big(\lambda_{x:A} s[\boldsymbol{\gamma}]\big)\{\alpha\} =_{\mathrm{def}} s[\boldsymbol{\gamma}, \alpha].$$

This satisfies the coherence condition for realizers of terms of dependent product type since $[\![f]\!]$ satisfies the corresponding coherence property. To see that $[\![\lambda_{x:A} s]\!]$ is functorial note that

$$\lambda_{x:A} s[\boldsymbol{\gamma}] \cdot (\boldsymbol{h}, \lambda_{x:A} s|\boldsymbol{h})\{\alpha\} = \lambda_{x:A} s[\boldsymbol{\gamma}]\big\{\alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})\big\} \cdot \big(\boldsymbol{h}, \iota(\boldsymbol{h})_a, \zeta\big)$$

$$= s[\boldsymbol{\gamma}, \alpha \cdot (\boldsymbol{h}^{-1}, 1_{a \cdot \boldsymbol{h}^{-1}})] \cdot (\boldsymbol{h}, \iota(\boldsymbol{h})_a, s|(\boldsymbol{h}, \iota(\boldsymbol{h})_a)$$

$$= s[\boldsymbol{\gamma}', \alpha]$$

$$= \lambda_{x:A} s[\boldsymbol{\gamma}']\{\alpha\},$$

where ζ is the morphism defined in (11) and where the third equation is by functoriality of $[\![s]\!]$.

Finally, suppose we are given a term $f : \prod_{x:A} B(x)$ and a term $a : A$, both in context Γ. Then we need to give the application term. For this, suppose given $(\boldsymbol{c}, \boldsymbol{\gamma})$ in $[\![\Gamma]\!]$, then we have

$$f[\boldsymbol{\gamma}]\{a[\boldsymbol{\gamma}]\} \Vdash_{\boldsymbol{\gamma}, a[\boldsymbol{\gamma}]} \mathtt{app}(f(\boldsymbol{c}), a(\boldsymbol{c})) : B(\boldsymbol{c}, a(\boldsymbol{c}))$$

and so we define

$$\mathtt{app}(f, a)[\boldsymbol{\gamma}] =_{\mathrm{def}} f[\boldsymbol{\gamma}]\{a[\boldsymbol{\gamma}]\}.$$

That this definition is functorial is then immediate.

Given these definitions, the equations which are required to hold for these application and abstraction terms are straightforwardly verified. $\qquad\square$

4.3. **Semantic identity types.** We now turn to the interpretation of identity types. Since we work in the 1-truncated version of the type theory, realizers of terms of identity type will be unique when they exist, and merely serve as a characteristic function for the graph of the functorial action on realizers, in the sense that a term $f : \underline{A}(a,b)$ is realized (relative to realizers $\alpha \Vdash a : A, \beta \Vdash b : A$) if and only if $\alpha \cdot f = \beta$. The only complication is the elimination rule, which requires us to construct a realizer for J-terms. This is taken care of by using the fact that there exists a morphism from $\phi(a)$ to the elimination term $\mathsf{J}(\phi, a, b, f)$; we may then use the functorial action on realizers to turn a realizer of $\phi(a)$ into one of $\mathsf{J}(\phi, a, b, f)$.

4.3.1. *Identity type formation.* Consider a semantic context $[\![\Gamma]\!] \to |\Gamma|$, and a type $(\Gamma, x : A, \Vdash)$ in context Γ, giving a context extension $[\![\Gamma, x : A]\!]$. In order to describe the identity type associated to this data, we will first given an explicit description of the groupoid $\|\Gamma, x, y : A, z : \underline{A}(x, y)\|$.

Objects: An object is a tuple $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f)$ such that
- $(\boldsymbol{c}, \boldsymbol{\gamma})$ is an object of $[\![\Gamma]\!]$;
- a, b are objects of the groupoid $|A(\boldsymbol{c})|$;
- $\alpha \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c})$;
- $\beta \Vdash_{\boldsymbol{\gamma}} b : A(\boldsymbol{c})$; and
- $f : \underline{A(\boldsymbol{c})}(a, b)$.

Note that it is not required that $\alpha \cdot f = \beta$.

Arrows: An arrow $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f) \to (\boldsymbol{c}', \boldsymbol{\gamma}', a', \alpha', b', \beta', f')$ is a tuple $(\boldsymbol{h}, m, n, l)$ such that
- $\boldsymbol{h} : (\boldsymbol{c}, \boldsymbol{\gamma}) \to (\boldsymbol{c}', \boldsymbol{\gamma}')$ is a morphism in $[\![\Gamma]\!]$;
- $m : a \cdot \boldsymbol{h} \to a'$ in is a morphism in $|A(\boldsymbol{c})|$ such that

$$\alpha \cdot (\boldsymbol{h}, m) = \alpha' \Vdash_{\boldsymbol{\gamma}'} a' : A(\boldsymbol{c}');$$

- n is an arrow

$$b \cdot (\boldsymbol{h}, m) \;=\; |\Gamma, x : A \;\vdash\; A|_{(\boldsymbol{h}, m)}(b) \to b'$$

in $|A(\boldsymbol{c})|$ such that

$$\beta \cdot (\boldsymbol{h}, m, n) = \beta' \Vdash_{\boldsymbol{\gamma}'} b' : A(\boldsymbol{c}'),$$

where we note that

$$\beta \cdot (\boldsymbol{h}, m, n) \;=\; [\![\Gamma, x : A \vdash A]\!]_{(\boldsymbol{h}, m, n)}(\beta) \;=\; [\![\Gamma \vdash A]\!]_{\hat{\tau}_{\Gamma; x:A; A}(\boldsymbol{h}, m, n)}(\beta); \text{ and}$$

- l is a propositional equality witnessing the equation (via 1-truncation these are the same)

$$f \cdot (\boldsymbol{h}, m, n) \;=\; |\Gamma, x, y : A \vdash \underline{A}(x, y)|_{(\boldsymbol{h}, m, n)}(f) = f'.$$

Next, we define the functor $[\![\Gamma, x, y : A \vdash \underline{A}(x, y)]\!]$ assigning realizers to identity terms. Given an object $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f)$, we set

$$\Vdash_{\boldsymbol{\gamma}, \alpha, \beta} f : \underline{A(\boldsymbol{c})}(a, b) \quad \text{iff} \quad \alpha \cdot (1_{\boldsymbol{c}}, f) = \beta.$$

Thus, regarded as a presheaf, this functor is a *subterminal* presheaf, i.e. a subpresheaf of the presheaf with constant value 1. When we need a name for the (by definition unique) realizer of f we shall use the symbol $\star$. Note that when f is realized, then in particular

$$(1_{\boldsymbol{c}}, f) : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha) \to (\boldsymbol{c}, \boldsymbol{\gamma}, b, \beta)$$

is an arrow in $[\![\Gamma, x : A]\!]$.

We must check functoriality, which in this case amounts to showing that for an arrow

$$(\boldsymbol{h}, m, n, l) : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f) \to (\boldsymbol{c}', \boldsymbol{\gamma}', a', \alpha', b', \beta', f')$$

in $\|\Gamma, x, y : A \vdash \underline{A}(x, y)\|$,

$$\Vdash_{\boldsymbol{\gamma}, \alpha, \beta} f : \underline{A(\boldsymbol{c})}(a, b) \quad \text{implies} \quad \Vdash_{\boldsymbol{\gamma}', \alpha', \beta'} f' : \underline{A(\boldsymbol{c}')}(a', b').$$

So assume that $\Vdash_{\boldsymbol{\gamma}, \alpha, \beta} f : \underline{A(\boldsymbol{c})}(a, b)$ holds. First, we observe that the diagram

in $\|\Gamma \vdash \underline{A}\|$ commutes. Thus, to see that $\Vdash_{\boldsymbol{\gamma}', \alpha', \beta'} f' : \underline{A(\boldsymbol{c})}(a', b')$ we reason as follows:

$$
\begin{aligned}
\alpha' \cdot (1_{\boldsymbol{c}'}, f') &= \big(\alpha \cdot (\boldsymbol{h}, m)\big) \cdot (1_{\boldsymbol{c}'}, f') \\
&= \big(\alpha \cdot (1_{\boldsymbol{c}}, f)\big) \cdot \hat{\tau}_{\Gamma; x:A; A}(\boldsymbol{h}, m, n) \\
&= \beta \cdot \hat{\tau}_{\Gamma; x:A; A}(\boldsymbol{h}, m, n) \\
&= \beta'.
\end{aligned}
$$

Finally, we must verify that this construction is stable under reindexing along context morphisms. This amounts to verifying that, for a context morphism $[\![\boldsymbol{m}]\!] : [\![\Delta]\!] \to [\![\Gamma]\!]$, the diagram

commutes. This, however, is an immediate consequence of the definition of the realizability relation and the fact that the horizontal map induced by the context morphism $[\![\boldsymbol{m}]\!]$ preserves the functorial action on the realizers.

4.3.2. *Introduction rule.* Consider next the term formation rule

$$\frac{\Gamma \vdash A}{\Gamma, x : A \vdash \mathbf{r}(x) : \underline{A}(x, x)}$$

Given an object $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha)$ of $[\![\Gamma, x : A]\!]$ we must show that

$$\Vdash_{\boldsymbol{\gamma}, \alpha} \mathbf{r}(a) : \underline{A(\boldsymbol{c})}(a, a).$$

This holds if and only if

$$\Vdash_{\boldsymbol{\gamma}, \alpha, \alpha} \mathbf{r}(a) : \underline{A(\boldsymbol{c})}(a, a)$$

which holds if and only if $\alpha \cdot (1_{\boldsymbol{c}}, \mathbf{r}(a)) = \alpha$. But the latter is trivially true.

27

4.3.3. *Elimination rule.* Next, we treat the elimination rule

$$\frac{\begin{array}{c}\Gamma, x, y : A, z : \underline{A}(x,y) \;\vdash\; B(x,y,z)\\ \Gamma, x : A \;\vdash\; \varphi : B(x,x,\mathtt{r}(x))\end{array}}{\Gamma, x, y : A, z : \underline{A}(x,y) \;\vdash\; \mathtt{J}(\varphi,x,y,z) : B(x,y,z)}$$

We assume given the interpretation of the judgement $\Gamma, x, y : A, z : \underline{A}(x,y) \;\vdash\; B(x,y,z)$, and, given an object $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star)$ of $[\![\Gamma, x, y : A, z : \underline{A}(x,y)]\!]$, a realizer $\varphi[\boldsymbol{\gamma}, \alpha] \Vdash_{\boldsymbol{\gamma},\alpha} \varphi(a) : B(a,a,\mathtt{r}(a))$.

We need to define a realizer $\xi \Vdash_{\boldsymbol{\gamma},\alpha,\beta,\star} \mathtt{J}(\varphi,a,b,f) : B(a,b,f)$. Actually, we shall define a functor which sends realizers $\varphi[\boldsymbol{\gamma}, \alpha]$ to realizers ξ as above.

So consider an object $(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star)$ of $[\![\Gamma, x, y : A, z : \underline{A}(x,y)]\!]$. We begin by constructing a map

$$(\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, a, \alpha, 1_a, \star) \to (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star)$$

in $[\![\Gamma, x, y : A, z : \underline{A}(x,y)]\!]$. For this, observe that there is a term

$$\mathtt{J}(\mathtt{r}(\mathtt{r}(x)), a, b, f) : \underline{A(\boldsymbol{c})}(a,b)\big(1_a \cdot (1_a, f), f\big)$$

and therefore a morphism

$$(1_{\boldsymbol{c}}, 1_a, f, \mathtt{J}(\mathtt{r}(\mathtt{r}(x)), a, b, f)) : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, a, \alpha, 1_a) \to (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f)$$

in $\|x, y : A, z : \underline{A}(x,y)\|$, where we have used the fact that

$$\begin{aligned}\alpha \cdot (1_{\boldsymbol{c}}, 1_a, f) &= \alpha \cdot \hat{\tau}_{\Gamma;x:A;A}(1_{\boldsymbol{c}}, 1_a, f)\\ &= \alpha \cdot (1_{\boldsymbol{c}}, f)\\ &= \beta.\end{aligned}$$

So we have constructed a morphism

$$(1_{\boldsymbol{c}}, 1_a, f, \mathtt{J}(\mathtt{r}(\mathtt{r}(x)), a, b, f)) : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, a, \alpha, 1_a, \star) \to (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star).$$

Then we observe that

$$\mathtt{J}\big(\mathtt{r}(\varphi(x)), a, b, f\big) : \underline{B(a,b,f)}\big(\varphi(a) \cdot \big(1_{\boldsymbol{c}}, 1_a, f, \mathtt{J}(\mathtt{r}(\mathtt{r}(x)), a, b, f)\big), \mathtt{J}(\varphi, a, b, f)\big),$$

and therefore we have an arrow

$$f \;\triangleright\; \varphi : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, a, \alpha, 1_a, \star, \varphi(a)) \to (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star, \mathtt{J}(\varphi, a, b, f))$$

in $\|\Gamma, x, y : A, z : \underline{A}(x,y), v : B(x,y,z)\|$ given by

$$f \;\triangleright\; \varphi =_{\mathrm{def}} (1_{\boldsymbol{c}}, 1_a, f, \mathtt{J}(\mathtt{r}(\mathtt{r}(x)), a, b, f), \mathtt{J}(\mathtt{r}(\varphi(x)), a, b, f)).$$

Therefore,

$$\varphi[\boldsymbol{\gamma}, \alpha] \cdot (f \;\triangleright\; \varphi) \Vdash_{\boldsymbol{\gamma},\alpha,\beta,\star} \mathtt{J}(\varphi, a, b, f) : B(\boldsymbol{c}, a, b, f)$$

and we define

$$\mathtt{J}(\varphi, a, b, f)[\boldsymbol{\gamma}, \alpha, \beta, \psi] =_{\mathrm{def}} \varphi[\boldsymbol{\gamma}, \alpha] \cdot (f \;\triangleright\; \varphi).$$

It remains to be seen that this is functorial. Suppose given an arrow

$$\theta = (\boldsymbol{h}, m, n, l) : (\boldsymbol{c}, \boldsymbol{\gamma}, a, \alpha, b, \beta, f, \star) \to (\boldsymbol{c}', \boldsymbol{\gamma}', a', \alpha', b', \beta', f', \star).$$

Then to show functoriality we need to prove that

$$\mathtt{J}(\varphi, a, b, f)[\boldsymbol{\gamma}, \alpha, \beta, \star] \cdot \big(\theta, \mathtt{J}(\varphi, a', b', f')|(\boldsymbol{h}, m, n, l)\big) = \mathtt{J}(\varphi, a', b', f')[\boldsymbol{\gamma}', \alpha', \beta', \star]. \qquad (12)$$

Well, we first observe that there is a commutative square

$$
\begin{array}{ccc}
(\boldsymbol{c}, a, a, 1_a, \varphi(a)) & \xrightarrow{(\boldsymbol{h},m,m',m'',\varphi|(\boldsymbol{h},m,m',m''))} & (\boldsymbol{c}', a', a', 1_{a'}, \varphi(a')) \\
{\scriptstyle f \,\triangleright\, \varphi} \downarrow & & \downarrow {\scriptstyle f' \,\triangleright\, \varphi} \\
(\boldsymbol{c}, a, b, f, \mathtt{J}(\varphi, a, b, f)) & \xrightarrow[(\theta, \mathtt{J}(\varphi,a,b,f)|\theta)]{} & (\boldsymbol{c}', a', b', f', \mathtt{J}(\varphi, a', b', f'))
\end{array}
\tag{13}
$$

where m' is the composite, arising via weakening,

$$
a \cdot (\boldsymbol{h}, m) \xrightarrow{a\dagger(\boldsymbol{h},m)} a \cdot \boldsymbol{h} \xrightarrow{m} a
$$

and similarly m'' is

$$
1_a \cdot (\boldsymbol{h}, m, m') \xrightarrow{1_a\dagger(\boldsymbol{h},m,m')} 1_a \cdot (\boldsymbol{h}, m) =\!\!=\!\!= 1_{a'}.
$$

Then, writing $\theta' = (\boldsymbol{h}, m, m', m'')$, we calculate

$$
\begin{aligned}
\mathtt{J}(\varphi, a, b, f)[\boldsymbol{\gamma}, \alpha, \beta, \star] \cdot \big(\theta, \mathtt{J}(\varphi, a, b, f)|\theta\big) &= \varphi[\boldsymbol{\gamma}, \alpha] \cdot (f \;\triangleright\; \varphi) \cdot (\theta, \mathtt{J}(\varphi, a, b, f)|\theta) \\
&= \varphi[\boldsymbol{\gamma}, \alpha] \cdot (\theta', \varphi|\theta') \cdot (f' \;\triangleright\; \varphi) \\
&= \varphi[\boldsymbol{\gamma}, \alpha] \cdot (\boldsymbol{h}, m, \varphi|(\boldsymbol{h}, m))) \\
&= \varphi[\boldsymbol{\gamma}', \alpha'] \cdot (f' \;\triangleright\; \varphi) \\
&= \mathtt{J}(\varphi, a', b', f')[\boldsymbol{\gamma}', \alpha', \beta', \star].
\end{aligned}
$$

Here, the first and last equalities are by definition of the realizers; the second equality is by virtue of the fact that (13) commutes; the third equality is by the definition of weakening, and the fourth is by functoriality of the action on realizers.

4.3.4. *Conversion.* The conversion rule

$$
\overline{\quad \Gamma, x : A \;\vdash\; \mathtt{J}(\varphi, x, x, \mathtt{r}(x)) = \varphi \quad}
$$

is trivially seen to be satisfied using the definitions given above.

This concludes the proof of:

Proposition 4.5. *The CwF* **Real**$(\mathcal{R})$ *supports identity types.*

4.4. **Semantic dependent sums.** Consider a context $[\![\Gamma]\!]$, a type A over this context with comprehension $[\![\Gamma, x : A]\!]$ and furthermore a type B over this comprehension, giving a further comprehension $[\![\Gamma, x : A, y : B(x)]\!]$. Thus we have a diagram

$$
\begin{array}{ccccc}
[\![\Gamma, x : A, y : B(x)]\!] & \longrightarrow & [\![\Gamma, x : A]\!] & \longrightarrow & [\![\Gamma]\!] \\
\downarrow & & \downarrow & & \downarrow \\
|\Gamma, x : A, y : B(x)| & \longrightarrow & |\Gamma, x : A| & \longrightarrow & |\Gamma|.
\end{array}
$$

We wish to define the semantic dependent sum type in context $[\![\Gamma]\!]$. To this end, we first form the syntactic type $\Gamma \vdash \sum_{x:A} B(x)$, giving rise to the pullback diagram

$$
\begin{array}{ccc}
\|\Gamma, v : \sum_{x:A} B(x)\| & \longrightarrow & |\Gamma, v : \sum_{x:A} B(x)| \\
\downarrow & & \downarrow \\
[\![\Gamma]\!] & \longrightarrow & |\Gamma|.
\end{array}
$$

We then must specify the realizability presheaf

$$\|\Gamma, v : \sum_{x:A} B(x)\| \to \mathbf{Set}.$$

For $(\boldsymbol{c}, \boldsymbol{\gamma})$ an object of $[\![\Gamma]\!]$, we define

$$v \Vdash_{\boldsymbol{\gamma}} p : \sum_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)$$

if and only if v is a pair (v_0, v_1) such that

$$v_0 \Vdash_{\boldsymbol{\gamma}} \pi_0(p) : A(\boldsymbol{c}), \text{ and}$$
$$v_1 \Vdash_{\boldsymbol{\gamma}, v_0} \pi_1(p) : B(\boldsymbol{c}, \pi_0(p)).$$

4.4.1. *Reindexing.* Suppose we are given a map $(\boldsymbol{h}, m) : (\boldsymbol{c}, \gamma, p) \to (\boldsymbol{c}', \gamma', p')$ in the groupoid $\|\Gamma \vdash \sum_{x:A} B(x)\|$ together with a realizer $v \Vdash_{\boldsymbol{\gamma}} p : \sum_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)$. Then we have

$$v_0 \cdot (\boldsymbol{h}, \xi_0) \Vdash_{\boldsymbol{\gamma}'} \pi_0(p') : A(\boldsymbol{c}')$$

where ξ_0 is the composite

$$\pi_0(p) \cdot \boldsymbol{h} \xrightarrow{\mathsf{J}^\sigma(\mathbf{r}\pi_0(v), \boldsymbol{c}, \boldsymbol{c}', \boldsymbol{h}, p)} \pi_0(p \cdot \boldsymbol{h}) \xrightarrow{\mathsf{J}(\mathbf{r}\pi_0(x), p \cdot \boldsymbol{h}, p', m)} \pi_0(p')$$

Similarly, we define ξ_1 to be the composite

$$\pi_1(p) \cdot (\boldsymbol{h}, \xi_0) \xrightarrow{\xi'} \pi_1(p \cdot \boldsymbol{h}) \cdot (1_{\boldsymbol{c}'}, \mathsf{J}(\mathbf{r}\pi_0 x, p \cdot \boldsymbol{h}, p', m)) \xrightarrow{\mathsf{J}(\mathbf{r}\pi_0 x, p \cdot \boldsymbol{h}, p', m))} \pi_1(p')$$

where $\xi' = \mathsf{J}^\sigma(\mathbf{r}\pi_1(v) \cdot (1_{\boldsymbol{x}}, \mathsf{J}(\mathbf{r}\pi_0 x, v, w, u)), \boldsymbol{c}, \boldsymbol{c}', p)$.

Then we have that

$$v_1 \cdot (\boldsymbol{h}, \xi_0, \xi_1) \Vdash_{\boldsymbol{\gamma}', v_0 \cdot (\boldsymbol{h}, \xi_0)} \pi_1(p') : B(\boldsymbol{c}', \pi_0(p'))$$

and we define

$$v \cdot (\boldsymbol{h}, m) =_{\mathrm{def}} \big(v_0 \cdot (\boldsymbol{h}, \xi_0), v_1 \cdot (\boldsymbol{h}, \xi_0, \xi_1)\big).$$

Functoriality of reindexing is a routine verification using the truncation rule.

Next, we must show that the formation of the dependent sum type is stable under substitution. It suffices to show that, for a context morphism $m : \Delta \to \Gamma$, the diagram

is commutative. To this end, consider an object $(\boldsymbol{d}, \boldsymbol{\delta}, p)$ and note that we have

$$v \Vdash_{m[\boldsymbol{\delta}]} p : \sum_{x:A[m]} B[m] \quad \Leftrightarrow \quad v_0 \Vdash_{m[\boldsymbol{\delta}]} \pi_0(p) : A[m] \text{ and } v_1 \Vdash_{m[\boldsymbol{\delta}], v_0} B[m](\pi_0(p))$$

$$\Leftrightarrow \quad v_0 \Vdash_{\boldsymbol{\delta}} \pi_0(p) : A \text{ and } v_1 \Vdash_{\boldsymbol{\delta}, v_0} \pi_1(p) : B(\pi_0(p))$$

where we have used Lemma 4.2 regarding the behavior of realizers under substitution.

4.4.2. *Introduction rule.* For the introduction rule

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : B(a)}{\Gamma \vdash \mathtt{pair}(a,b) : \sum_{x:A} B(x)}$$

we reason as follows. For an object $(\boldsymbol{c}, \boldsymbol{\gamma})$ of $[\![\Gamma]\!]$ we define

$$\mathtt{pair}(a,b)[\boldsymbol{\gamma}] =_{\mathrm{def}} (a[\boldsymbol{\gamma}], b[\boldsymbol{\gamma}]).$$

For the coherence property note that, given $\boldsymbol{h} : (\boldsymbol{c}, \boldsymbol{\gamma}) \to (\boldsymbol{c}', \boldsymbol{\gamma}')$, we have

$$\mathtt{pair}(a,b)[\boldsymbol{\gamma}] \cdot (\boldsymbol{h}, \mathtt{pair}(a,b)|\boldsymbol{h}) = (a[\boldsymbol{\gamma}], b[\boldsymbol{\gamma}]) \cdot (\boldsymbol{h}, \mathtt{pair}(a,b)|\boldsymbol{h})$$
$$= \big(a[\boldsymbol{\gamma}] \cdot (\boldsymbol{h}, \xi_0), b[\boldsymbol{\gamma}] \cdot (\boldsymbol{h}, \xi_0, \xi_1)\big).$$

Since, in this case, the truncation rule gives that $\xi_0 = a|\boldsymbol{h}$ and $\xi_1 = b|(\boldsymbol{h}, \xi_0)$, the required equation is then immediate, and it also follows without difficulty that this term formation operation commutes with substitution with respect to context morphisms.

4.4.3. *Elimination rule.* For the elimination rule

$$\frac{\Gamma, z : \sum_{x:A} B(x) \vdash C(z) \qquad \Gamma, x : A, y : B(x) \vdash \psi : C(\mathtt{pair}(x,y))}{\Gamma, z : \sum_{x:A} B(x) \vdash \mathtt{R}(\psi, z) : C(z)}$$

we assume given $(\boldsymbol{c}, \boldsymbol{\gamma}, p, \upsilon)$ in $[\![\Gamma, z : \sum_{x:A} B(x)]\!]$. Then

$$\psi[\boldsymbol{\gamma}, \upsilon_0, \upsilon_1] \Vdash_{\boldsymbol{\gamma}, \upsilon_0, \upsilon_1} \psi(\boldsymbol{c}, \pi_0 p, \pi_1 p) : C(\boldsymbol{c}, \mathtt{pair}(\pi_0 p, \pi_1 p)),$$

and so

$$\psi[\boldsymbol{\gamma}, \upsilon_0, \upsilon_1] \Vdash_{\boldsymbol{\gamma}, \upsilon} \psi(\boldsymbol{c}, \pi_0 p, \pi_1 p) : C(\boldsymbol{c}, \mathtt{pair}(\pi_0 p, \pi_1 p)).$$

Now, we have the term

$$\Gamma, z : \sum_{x:A} B(x) \vdash \underline{\sum_{x:A} B(x)}\big(\mathtt{pair}(\pi_0 z, \pi_1 z), z\big)$$

$$\frac{\Gamma, x : A, y : B(x) \vdash \mathtt{r}(\mathtt{pair}(x,y)) : \underline{\sum_{x:A} B(x)}\big(\mathtt{pair}(x,y), \mathtt{pair}(x,y)\big)}{\vdash \mathtt{R}(\mathtt{r}(\mathtt{pair}(x,y)), p) : \underline{\sum_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)}\big(\mathtt{pair}(\pi_0 p, \pi_1 p), p\big)}$$

Now, it is easily seen, by inspecting the definition of the reindexing action on realizers for dependent sums, that in fact we have an arrow

$$(1_{\boldsymbol{c}}, \mathtt{R}(\mathtt{r}(\mathtt{pair}(x,y), p))) : (\boldsymbol{c}, \boldsymbol{\gamma}, \mathtt{pair}(\pi_0 p, \pi_1 p), \upsilon) \to (\boldsymbol{c}, \boldsymbol{\gamma}, p, \upsilon)$$

in $[\![\Gamma \vdash \sum_{x:A} B(x)]\!]_{(\boldsymbol{c}, \boldsymbol{\gamma}, p)}$. Therefore,

$$\psi[\boldsymbol{\gamma}, \upsilon_0, \upsilon_1] \cdot (1_{\boldsymbol{c}}, \mathtt{R}(\mathtt{rpair}(x,y), p), \mathtt{R}(\mathtt{r}\psi(\boldsymbol{c}, \pi_0 z, \pi_1 z), p) \Vdash_{\boldsymbol{\gamma}, \upsilon} \mathtt{R}(\psi, p) : C(\boldsymbol{c}, p).$$

Here the term $\mathtt{R}(\mathtt{r}\psi(\boldsymbol{c}, x, y), p)$ is constructed as follows:

$$z : \sum_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x) \vdash \underline{C(\boldsymbol{c}, z)}\big(\psi(\boldsymbol{c}, \pi_0 z, \pi_1 z) \cdot (1_{\boldsymbol{c}}, \mathtt{R}(\mathtt{r}(\mathtt{pair}(x,y)), z)), \mathtt{R}(\psi(\boldsymbol{c}, x, y), z)\big)$$

$$\frac{x : A, y : B(x) \vdash \mathtt{r}\psi(\boldsymbol{c}, x, y) : \underline{C(\boldsymbol{c}, \mathtt{pair}(x,y))}\big(\psi(\boldsymbol{c}, x, y), \psi(\boldsymbol{c}, x, y)\big)}{\vdash \mathtt{R}(\mathtt{r}\psi(\boldsymbol{c}, x, y), p) : \underline{C(\boldsymbol{c}, p)}\big(\psi(\boldsymbol{c}, \pi_0 p, \pi_1 p) \cdot (1_{\boldsymbol{c}}, \mathtt{R}(\mathtt{r}(\mathtt{pair}(x,y)), p)), \mathtt{R}(\psi(\boldsymbol{c}, x, y), p)\big).}$$

So we define

$$\mathtt{R}(\psi, p)[\boldsymbol{\gamma}, \upsilon] =_{\mathrm{def}} \psi[\boldsymbol{\gamma}, \upsilon_0, \upsilon_1] \cdot (1_{\boldsymbol{c}}, \mathtt{R}(\mathtt{rpair}(x,y), p), \mathtt{R}(\mathtt{r}\psi(\boldsymbol{c}, \pi_0 z, \pi_1 z), p)$$

4.4.4. *Conversion rule.* The conversion rule

$$\overline{\Gamma, x : A, y : B(x) \ \vdash\ \mathtt{R}(\psi, \mathtt{pair}(x,y)) \ =\ \psi(x,y) : C(\mathtt{pair}(x,y))}$$

is trivial given the definitions above.

This concludes the proof of:

Proposition 4.6. *The CwF* **Real**$(\mathcal{R})$ *supports dependent sums.*

4.5. **Semantic natural numbers.** Although it is in principle possible to consider different interpretations of the type of natural numbers (when suitable notions of realizability are provided) we will simply fix a uniform interpretation of natural numbers. Observe that, in the groupoid model built over the free groupoid on G the type of natural numbers $\mathtt{N}$ is interpreted as the discrete groupoid of natural numbers. In particular, in this interpretation, each $t : \mathtt{N}$ is sent to a genuine (external) natural number n_t. This n_t can be represented by theoretically by the corresponding number, which we denote by $\#(t)$. Thus, for $t : \mathtt{N}$, we have a numeral $\#(t) : \mathtt{N}$. By the construction of the groupoid model, this gives an endofunctor $\# : |\mathtt{N}| \to |\mathtt{N}|$ and we define

$$\tau \Vdash t : \mathtt{N} \quad \text{if and only if } \tau : \underline{\mathtt{N}}(t, \#t).$$

Note that for any $f : \underline{\mathtt{N}}(s, t)$ we have $\#f = \mathtt{r}(\#s)$ since $\mathtt{N}$ is interpreted as a discrete groupoid under the groupoid interpretation.

By the functoriality of $\#$ this determines the interpretation $[\![\mathtt{N}]\!]$ of the type of natural numbers in **Real**$(\mathcal{R})$. Observe that $\#(n) = n$ when n is a numeral. The successor operation gives an endofunctor $|\mathtt{S}| : |\mathtt{N}| \to |\mathtt{N}|$ and, by inspection of the groupoid interpretation, this functor commutes with $\#$ in the sense that $\# \circ |\mathtt{S}| = |\mathtt{S}| \circ \#$. Using these facts it is possible to describe the interpretations of the terms coming from the introduction rules. First, we take $[\![0]\!]$ to be given by the realizer $\mathtt{r}(0) \Vdash 0 : \mathtt{N}$. Next, given $\tau \Vdash t : \mathtt{N}$, we have, by the aforementioned facts, that $|S|(\tau) \Vdash S(t) : \mathtt{N}$. As such, we take this term to be the realizer part of the interpretation of $[\![\mathtt{S}(t)]\!]$. The interpretation of the elimination rule will require a little bit more care.

4.5.1. *Elimination rule.* We now will show that the elimination rule

$$\frac{\Gamma \ \vdash\ b : B(0) \qquad \Gamma, x : \mathtt{N}, y : B(x) \ \vdash\ g(x,y) : B\big(\mathtt{S}(x)\big)}{\Gamma, n : \mathtt{N} \ \vdash\ \mathtt{rec}(b, g, n) : B(n)}$$

is valid in the interpretation.

Let $(\boldsymbol{c}, \boldsymbol{\gamma}, n, \nu)$ in $[\![\Gamma, n : \mathtt{N}]\!]$ be given. In what follows we will suppress $\boldsymbol{c}$ as much as possible in order to avoid notational clutter.

We will construct the interpretation of the elimination term by induction on $\#n$. In the base case we take $\mathtt{rec}(b, g)[\boldsymbol{\gamma}, \nu] \Vdash_{\boldsymbol{\gamma}, \nu} \mathtt{rec}(b, g, n) : B(\boldsymbol{c}, n)$ to be given by

$$b[\boldsymbol{\gamma}] \cdot \big(\mathtt{rec}(b, g) | (1_{\boldsymbol{c}}, \nu^{-1})\big) \Vdash_{\boldsymbol{\gamma}, \nu} \mathtt{rec}(n, b, g) : B(\boldsymbol{c}, n).$$

When $\#n = \mathtt{S}(\mathtt{m})$ for $\mathtt{m}$ a numeral, we have by induction hypothesis,

$$\mathtt{rec}(b, g)[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})] \Vdash_{\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})} \mathtt{rec}(b, g, \mathtt{m}) : B(\boldsymbol{c}, \mathtt{m}).$$

By the conversion rule $\mathtt{rec}(b, g, \mathtt{S}(\mathtt{m})) = g(\mathtt{m}, \mathtt{rec}(b, g, \mathtt{m}))$ and therefore

$$g\big[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m}), \mathtt{rec}(b, g)[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})]\big] \Vdash_{\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m}), \mathtt{rec}(b,g)[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})]} \mathtt{rec}(b, g, \mathtt{S}(\mathtt{m})) : B(\boldsymbol{c}, \mathtt{S}(\mathtt{m})),$$

which holds if and only if

$$g\big[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m}), \mathtt{rec}(b, g)[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})]\big] \Vdash_{\boldsymbol{\gamma}, \mathtt{r}(\mathtt{S}(\mathtt{m}))} \mathtt{rec}(b, g, \mathtt{S}(\mathtt{m})) : B(\boldsymbol{c}, \mathtt{S}(\mathtt{m})).$$

As such, we may define

$$\mathtt{rec}(b, g, n)[\boldsymbol{\gamma}] =_{\mathrm{def}} g\big[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m}), \mathtt{rec}(b, g)[\boldsymbol{\gamma}, \mathtt{r}(\mathtt{m})]\big] \cdot \big(\mathtt{rec}(b, g) | (1_{\boldsymbol{c}}, \nu^{-1})\big) \Vdash_{\boldsymbol{\gamma}, \nu} \mathtt{rec}(n, b, g) : B(\boldsymbol{c}, n).$$

4.5.2. *Conversion rule.* The conversion rule $\mathtt{rec}(b, g, 0) = b$ is immediate. For the conversion rule

$$\Gamma, n : \mathtt{N} \ \vdash \ \mathtt{rec}\big(b, g, \mathtt{S}(n)\big) = g\big(n, \mathtt{rec}\big(b, g, n\big)\big) : B\big(\mathtt{S}(n)\big)$$

we proceed by induction on $\#n$. In the base case, where $\#n = 0$, we have

$$
\begin{aligned}
\mathtt{rec}(b, g)[\boldsymbol{\gamma}, |S|(\nu)] &= g\big[\boldsymbol{\gamma}, \mathtt{r}(0), b[\boldsymbol{\gamma}]\big] \cdot \big(\mathtt{rec}(b, g)|(1_{\boldsymbol{c}}, |S|(\nu)^{-1})\big) \\
&= g\big[\boldsymbol{\gamma}, \mathtt{r}(0), b[\boldsymbol{\gamma}]\big] \cdot \big(g|(1_{\boldsymbol{c}}, \nu^{-1}, \mathtt{rec}(b, g)|(1_{\boldsymbol{c}}, \nu^{-1}))\big) \\
&= g\big[\boldsymbol{\gamma}, \nu, b[\boldsymbol{\gamma}] \cdot (\mathtt{rec}(b, g)|(1_c, \nu^{-1}))\big] \\
&= g\big[\boldsymbol{\gamma}, \nu, \mathtt{rec}(b, g)[\boldsymbol{\gamma}, \nu]\big],
\end{aligned}
$$

where the penultimate equation is by functoriality of the interpretation of g and the equation

$$\mathtt{rec}(b, g)|(1_{\boldsymbol{c}}, |S|(\nu)^{-1}) = g|(1_{\boldsymbol{c}}, \nu^{-1}, \mathtt{rec}(b, g)|(1_{\boldsymbol{c}}, \nu^{-1}))$$

is by 1-truncation. In the case where $\#n = \mathtt{S(m)}$ for $\mathtt{m}$ the equation follows from the induction hypothesis and the same reasoning as in the base case.

This completes the proof of:

Proposition 4.7. *The CwF* **Real**$(\mathcal{R})$ *supports the type of natural numbers.*

Structural rules.

$$\frac{\Gamma \vdash \mathcal{J}}{\Delta, \Gamma \vdash \mathcal{J}} \; (\dagger) \qquad \frac{a : A \qquad x : A, \Delta \vdash B(x)}{\Delta[a/x] \vdash B(a)} \qquad \frac{a : A \qquad x : A, \Delta \vdash b(x) : B(x)}{\Delta[a/x] \vdash b(a) : B(a)} \qquad \frac{A}{x : A, \Delta \vdash x : A}$$

($\dagger$): $\mathcal{J}$ ranges over judgements and the variables declared in Δ and Γ are disjoint.

Equality rules.

$$\frac{A}{A = A} \qquad \frac{A = B}{B = A} \qquad \frac{A = B \quad B = C}{A = C} \qquad \frac{a : A}{a = a : A} \qquad \frac{a = b : A}{b = a : A} \qquad \frac{a = b : A \quad b = c : A}{a = c : A}$$

$$\frac{a = b : A \qquad x : A \vdash B(x)}{B(a) = B(b)} \qquad \frac{a = b : A \qquad x : A \vdash f(x) : B(x)}{f(a) = f(b) : B(a)} \qquad \frac{A = B \quad a : A}{a : B}$$

Formation rules.

$$\frac{x : A \vdash B(x)}{\prod_{x:A} B(x)} \qquad \frac{x : A \vdash B(x)}{\sum_{x:A} B(x)} \qquad \frac{a, b : A}{\vdash \underline{A}(a, b)} \qquad \frac{}{\vdash \mathtt{N}}$$

Introduction rules.

$$\frac{x : A \vdash f(x) : B(x)}{\lambda_{x:A} f(x) : \prod_{x:A} B(x)} \qquad \frac{a : A \quad b : B(a)}{\mathtt{pair}(a, b) : \sum_{x:A} B(x)} \qquad \frac{a : A}{r(a) : \underline{A}(a, a)} \qquad \frac{}{\mathtt{0} : \mathtt{N}} \qquad \frac{n : \mathtt{N}}{\mathtt{S}(n) : \mathtt{N}}$$

Elimination rules.

$$\frac{f : \prod_{x:A} B(x) \qquad a : A}{\mathtt{app}(f, a) : B(a)} \qquad \frac{\vdash p : \sum_{x:A} B(x) \qquad x : A, y : B(x) \vdash \psi(x, y) : C(\mathtt{pair}(x, y))}{\mathtt{R}([x : A, y : B(x)]\psi(x, y), p) : C(p)}$$

$$\frac{\begin{array}{c} x : A, y : A, z : \underline{A}(x, y) \vdash B(x, y, z) \\ x : A \vdash \varphi(x) : B(x, x, r(x)) \\ f : \underline{A}(a, b) \end{array}}{\mathtt{J}(\varphi, a, b, f) : B(a, b, f)} \qquad \frac{n : \mathtt{N} \quad b : B(\mathtt{0}) \quad x : \mathtt{N}, y : B(x) \vdash g(x, y) : B(\mathtt{S}(x))}{\mathtt{rec}(b, g, n) : B(n)}$$

Conversion rules.

$$\frac{\lambda_{x:A} f(x) : \prod_{x:A} B(x) \qquad a : A}{\mathtt{app}(\lambda_{x:A} f(x), a) \; = \; f(a) : B(a)} \qquad \frac{a : A \quad b : B(a) \quad x : A, y : B(x) \vdash \psi(x, y) : C(\mathtt{pair}(x, y))}{\mathtt{R}(\psi, \mathtt{pair}(a, b)) \; = \; \psi(a, b) : C(\mathtt{pair}(a, b))}$$

$$\frac{a : A}{\mathtt{J}(\varphi, a, a, r(a)) \; = \; \varphi(a) : B(a, a, r(a))} \qquad \frac{}{\mathtt{rec}(b, g, \mathtt{0}) = b : B(\mathtt{0})}$$

Appendix B. Syntactic constructions

B.1. **Parameterized J-terms.** We describe a parameterized version of the elimination rule for identity types. Consider a context Δ of the form

$$\Delta = \big(x, y : A, z : \underline{A}(x,y), v_1 : B_1(x,y,z), \ldots, v_n : B_n(x,y,z,\vec{v})\big).$$

Then the following rule gives the parameterized terms $\mathtt{J}(\varphi, a, b, f, \vec{v})$:

$$\frac{\begin{array}{c} \Delta \;\vdash\; T(x,y,z,\vec{v}) \\ \Delta[x/y, \mathtt{r}(x)/z] \;\vdash\; \varphi(x,\vec{v}) : T\big(x,x,r(x),\vec{v}\big) \\ f : \underline{A}(a,b) \end{array}}{\Delta[a/x, b/y, f/z] \;\vdash\; \mathtt{J}([x:A]\varphi, a, b, f, \vec{v}) : T(a,b,f,\vec{v})} \;\; \mathtt{Id} \text{ elimination}$$

The construction is by induction on the length of the list of parameters $\vec{v}$. When $n = 0$ we simply set $\mathtt{J}(\varphi, a, b, f, ()) = \mathtt{J}(\varphi, a, b, f)$. Next, assume we have constructed the terms $\mathtt{J}(\varphi, a, b, f, v_1, \ldots, v_m)$ for lists of parameters of length $m < n$. Assume we are given a judgement of the form

$$\Delta[x/y, \mathtt{r}(x)/z] \;\vdash\; \varphi(x, v_1, \ldots, v_n) : T(x, x, \mathtt{r}(x), v_1, \ldots, v_n).$$

We may then form the term

$$\lambda_{v_n : B_n(x,x,\mathtt{r}(x),\vec{v})} \varphi(x,\vec{v}) : \prod_{v_n : B_n(x,x,\mathtt{r}(x),\vec{v})} T(x,x,\mathtt{r}(x),\vec{v})$$

in context

$$\big(x : A, v_1 : B_1(x,x,\mathtt{r}(x)), \ldots, v_{n-1} : B_{n-1}(x,x,\mathtt{r}(x),\vec{v})\big).$$

Thus by induction hypothesis we have the parameterized term

$$\frac{\begin{array}{c} x, y : A, z : \underline{A}(x,y), v_1 : B_1(x,y,z), \ldots, v_{n-1} : B_{n-1}(x,y,z,\vec{v}) \;\vdash\; \prod_{v_n} T(x,y,z,\vec{v}) \\ \big(x : A, v_1 : B_1(x,x,\mathtt{r}(x)), \ldots, v_{n-1} : B_{n-1}(x,x,\mathtt{r}(x),\vec{v})\big) \;\vdash\; \lambda_{v_n}\varphi(x,\vec{v}) : \prod_{v_n} T(x,x,r(x),\vec{v}) \\ f : \underline{A}(a,b) \end{array}}{v_1 : B_1(a,b,f), \ldots, v_{n-1} : B_{n-1}(a,b,f,\vec{v}) \;\vdash\; \mathtt{J}(\lambda_{v_n}\varphi(x,\vec{v}), a, b, f, \vec{v}) : \prod_{v_n} T(a,b,f,\vec{v})}$$

Hence we may apply this term to v_n to obtain the desired term:

$$\mathtt{J}(\varphi, a, b, f, \vec{v}) =_{\text{def}} \mathtt{app}\big(\mathtt{J}(\lambda_{v_n}\varphi(x,\vec{v}), a, b, f, \vec{v}), v_n\big).$$

It is readily verified that these terms satisfy the conversion rule

$$\frac{\vdash a : A}{\Delta[a/x, a/y, \mathtt{r}(a)/z] \;\vdash\; \mathtt{J}(\varphi, a, a, \mathtt{r}(a), \vec{v}) = \varphi(a,\vec{v}) : T(a,a,\mathtt{r}(a),\vec{v})}$$

B.2. **Sequential J-terms.** We now turn to the construction of sequential J-terms, which arise in the setting of repeated Grothendieck constructions. We will actually need a parameterized version of these as well. That is, given a context

$$\Gamma = \big(x_1 : A_1, \ldots x_n : A_n(x_1, \ldots, x_{n-1})\big)$$

we consider an extended context

$$\Delta = \big(x_1 : A_1, \ldots x_n : A_n(x_1, \ldots, x_{n-1}), \vec{v}\big)$$

where the variables v_n are considered as parameters. Then we have the associated context

$$\tilde{\Delta} =_{\text{def}} \tilde{\Gamma}, \vec{v}.$$

We wish to establish the following derived rule:

$$\tilde{\Delta} \vdash T$$

$$\frac{\Delta \vdash \varphi : T[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}, \vec{v}]}{\tilde{\Delta} \vdash \mathtt{J}^\sigma([\boldsymbol{x}]\varphi, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \vec{v}) : T(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \vec{v})}$$

This is done by induction on the length of the context Γ. When $\Gamma = (x_1 : A_1)$, the problem reduces to the construction of the parameterized term $\mathtt{J}([x]\varphi, x, y, z, \vec{v})$, which was done in the previous section.

Thus assume that we have constructed the parameterized sequential terms for contexts of length $n - 1$. First consider the context

$$\begin{aligned}
\Delta' \quad =_{\mathrm{def}} \quad & \Delta[x_1/y_1, \mathtt{r}(x_1)/z_1, \ldots, x_{n-1}/y_{n-1}, \mathtt{r}(x_{n-1})/z_{n-1}] \\
= \quad & \big(x_1 : A_1, \ldots, x_{n-1} : A_{n-1}, x_n, y_n : A_n, z_n : \underline{A_n}(x_n, y_n), \vec{v}\big)
\end{aligned}$$

Then we have

$$\Delta' \vdash T(x_1, x_1, \mathtt{r}(x_1), \ldots, x_{n-1}, x_{n-1}, \mathtt{r}(x_{n-1}), x_n, y_n, z_n, \vec{v}).$$

Noting that $\Delta'[x_n/y_n, \mathtt{r}(x_n)/z_n] = \Delta$, we may thus use the ordinary parameterized elimination rule to form the judgement

$$\Delta' \vdash T(x_1, x_1, \mathtt{r}(x_1), \ldots, x_{n-1}, x_{n-1}, \mathtt{r}(x_{n-1}), x_n, y_n, z_n, \vec{v})$$

$$\frac{\Delta \vdash \varphi : T[\boldsymbol{x}/\boldsymbol{y}, \mathtt{r}(\boldsymbol{x})/\boldsymbol{z}, \vec{v}]}{\Delta' \vdash \mathtt{J}([x_n]\varphi, x_n, y_n, z_n, \vec{v}) : T((x_1, x_1, \mathtt{r}(x_1), \ldots, x_{n-1}, x_{n-1}, \mathtt{r}(x_{n-1}), x_n, y_n, z_n, \vec{v}))}$$

We can now apply the induction hypothesis to this term (where the parameters are now $x_n, y_n, z_n, \vec{v}$).

Appendix C. Realizability clauses

Here we collect for easy reference the realizability clauses extracted from the interpretation of type theory described above.

C.1. **Structural rules.**

Weakening. Assume we have judgements $\Gamma \vdash A$ and $\Delta \vdash$. In terms of realizers, given $(\boldsymbol{c}, \boldsymbol{\gamma}, \boldsymbol{d}, \boldsymbol{\delta})$ and object of $[\![\Gamma, \Delta]\!]$ and a term $a : A(\boldsymbol{c})$ we have

$$\alpha \Vdash_{\boldsymbol{\gamma}, \boldsymbol{\delta}} a : A(\boldsymbol{c}) \quad \text{iff} \quad \alpha \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c}).$$

For reindexing, given an arrow $(\boldsymbol{h}, \boldsymbol{k}, m) : (\boldsymbol{c}, \boldsymbol{\gamma}, \boldsymbol{d}, \boldsymbol{\delta}, a) \to (\boldsymbol{c}', \boldsymbol{\gamma}', \boldsymbol{d}', \boldsymbol{\delta}', a')$ in $\|\Gamma, \Delta \vdash A\|$, we have[2]

$$\alpha \cdot (\boldsymbol{h}, \boldsymbol{k}, m) = \alpha \cdot \hat{\tau}_{\Gamma;\Delta;A}(\boldsymbol{h}, \boldsymbol{k}, m).$$

For terms, given $\Gamma \vdash a : A$, we have

$$a[\boldsymbol{\gamma}, \boldsymbol{\delta}] = a[\boldsymbol{\gamma}].$$

---

[2]Explicitly, we have, in the notation of Section 3, that

$$\alpha \cdot (\boldsymbol{h}, \boldsymbol{k}, m) = \alpha \cdot \big(\boldsymbol{h}, m \circ (a\dagger(\boldsymbol{h}, \boldsymbol{k}))\big).$$

Substitution. Given

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x : A, \Delta \vdash B(x)}{\Gamma, \Delta[a/x] \vdash B(a)}$$

we have

$$\beta \Vdash_{\boldsymbol{\gamma}, \boldsymbol{\delta}} b : B(\boldsymbol{c}, a(\boldsymbol{c}), \boldsymbol{d}) \quad \text{iff} \quad \beta \Vdash_{\boldsymbol{\gamma}, a[\boldsymbol{\gamma}], \boldsymbol{\delta}} b : B(\boldsymbol{c}, a(\boldsymbol{c}), \boldsymbol{d})$$

and

$$\beta \cdot (\boldsymbol{h}, \boldsymbol{k}, m) = \beta \cdot \hat{\sigma}_{\Gamma; a; \Delta}(\boldsymbol{h}, \boldsymbol{k}, m)$$

for $(\boldsymbol{h}, \boldsymbol{k}, m) : (\boldsymbol{c}, \boldsymbol{\gamma}, \boldsymbol{d}, \boldsymbol{\delta}, b) \to (\boldsymbol{c}', \boldsymbol{\gamma}', \boldsymbol{d}', \boldsymbol{\delta}', b')$.

For terms, given

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x : A, \Delta \vdash b(x) : B(x)}{\Gamma, \Delta[a/x] \vdash b(a) : B(a)}$$

we have

$$b(a)[\boldsymbol{\gamma}, \boldsymbol{\delta}] = b[\boldsymbol{\gamma}, a[\boldsymbol{\gamma}], \boldsymbol{\delta}].$$

Dependent products. Given $\Gamma \vdash \prod_{x:A} B(x)$,

$$\phi \Vdash_{\boldsymbol{\gamma}} f : \prod_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)$$

if and only if ϕ is an operation

$$\bigl(\alpha \Vdash_{\boldsymbol{\gamma}} a : A(\boldsymbol{c})\bigr) \quad \longmapsto \quad \bigl(\phi\{\alpha\} \Vdash_{\boldsymbol{\gamma}, \alpha} \mathtt{app}(f, a) : B(\boldsymbol{c}, a)\bigr)$$

such that

$$\phi\{\alpha\} \cdot \bigl(1_{\boldsymbol{c}}, m, \mathtt{app}(f, x)|(1_{\boldsymbol{c}}, m)\bigr) = \phi\{\alpha \cdot (1_{\boldsymbol{c}}, m)\}$$

for $m : a \to a'$ in $|A(\boldsymbol{c})|$.

Dependent sums. Given $\Gamma \vdash \sum_{x:A} B(x)$,

$$\upsilon \Vdash_{\boldsymbol{\gamma}} p : \sum_{x:A(\boldsymbol{c})} B(\boldsymbol{c}, x)$$

if and only if $\upsilon = (\upsilon_0, \upsilon_1)$ with

$$\upsilon_0 \Vdash_{\boldsymbol{\gamma}} \pi_0(p) : A(\boldsymbol{c}), \text{ and}$$
$$\upsilon_1 \Vdash_{\boldsymbol{\gamma}, \upsilon_0} \pi_1(p) : B(\boldsymbol{c}, \pi_0(p)).$$

Identity types. Given $\Gamma, x : A, y : A \vdash \underline{A}(x, y)$,

$$\Vdash_{\boldsymbol{\gamma}, \alpha, \beta} f : \underline{A}(\boldsymbol{c})(a, b) \quad \text{iff} \quad \alpha \cdot (1_{\boldsymbol{c}}, f) = \beta.$$

Natural numbers. Where $\#(t) : \mathtt{N}$ denotes the numeral associated, under the groupoid interpretation built over the free groupoid, to the closed term $t : \mathtt{N}$, we have

$$\tau \Vdash t : \mathtt{N} \quad \text{iff} \quad \tau : \underline{\mathtt{N}}(t, \#t).$$

37

References

[1] S. Awodey, *Type theory and homotopy*, to appear, 2010.

[2] S. Awodey, P. Hofstra, and M. A. Warren, *Martin-löf Complexes*, submitted, 2009.

[3] P. Dybjer, *Internal type theory*, Proc. BRA TYPES workshop, Torino, June 1995 (Berlin), Lecture Notes in Comput. Sci., vol. 1158, Springer-Verlag, 1996.

[4] N. Gambino and R. Garner, *The identity type weak factorization system*, Theoretical Computer Science **409** (2008), no. 1, 94–109.

[5] M. Hofmann, *Syntax and semantics of dependent types*, Semantics and Logics of Computation (P. Dybjer and A. M. Pitts, eds.), Publications of the Newton Institute, Cambridge University Press, Cambridge, 1997, pp. 79–130.

[6] M. Hofmann and T. Streicher, *The groupoid interpretation of type theory*, Twenty-Five Years of Constructive Type Theory (G. Sambin and J. Smith, eds.), Oxford Logic Guides, vol. 36, Oxford University Press, Oxford, 1998, pp. 83–111.

[7] W. W. Tait, *Intensional interpretations of functionals of finite type I*, J. Symbolic Logic **32** (1967), no. 2, 198–212.

[8] V. Voevodsky, *Univalent foundations project*, Modified version of an NSF grant application, 2010.

Department of Mathematics and Statistics, University of Ottawa, 585 King Edward Ave., Ottawa, Ontario K1N 6N5 Canada

E-mail address: `phofstra@uottawa.ca`

School of Mathematics, Institute for Advanced Study, Einstein Dr., Princeton, New Jersey 08540 USA

E-mail address: `mwarren@math.ias.edu`